НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Інститут Прикладного Системного Аналізу

Математичних методів системного аналізу

«На правах рукопису»
УДК _519.8(075.8) 681.3.07_____

«До захисту допущено»
Завідувач кафедри
_____ Тимощук О.Л
«___»_____20__ р.

Магістерська дисертація

на здобуття ступеня магістра

зі спеціальності 102 Комп'ютерні науки

на тему: «Розпізнавання медичних зображень з використанням нечітких нейронних мереж**»**

Виконав:
студент VI курсу, групи КА-64м
Варга Ігор Юрійович _____

Керівник:

Професор, д.т.н,
Зайченко Ю.П

_____

Рецензент:
С.н.с, к.т.н,
Віштальд Д.М.

_____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка) _____

Київ 2018

РЕФЕРАТ

Магістерська дисертація: 86с., 18 рис., 27 табл., ** джерел та 2 додатки.

Об`єкт дослідження – медичні зображення тканей людини

Предмет дослідження – методи розпізнавання медичних зображень.

Мета роботи – побудувати класифікатор тканей органів людини, для класифікації злоякісних та доброякісних пухлин.

В роботі реалізовані алгоритми для класифікації зображень, з використанням попередньо навченої моделі для виділення признаків. Проведений порівняльний аналіз ефективності різних моделей.

Зроблені висновки щодо ефективності данних моделей, та рекомендації для розвитку продукту.

КЛАСИФІКАЦІЯ, РОСПІЗНАВАННЯ ОБРАЗІВ, АВТОМАТИЧНА ДІАГНОСТИКА, РОСПІЗНАВАННЯ МЕДИЧНИХ ЗОБРАЖЕНЬ, НЕЧІТКІ НЕЙРОННІ МЕРЕЖІ

ABSTRACT

Master thesis contains 86 p., 18 fig., 27 tabl., ** sources, 2 appendixes.

The object of the study is medical images of human tissues

Subject of research - methods of recognition of medical images.

The purpose of the work is to build a classifier of tissues of human organs, for the classification of malignant and benign tumors. In this work, algorithms are implemented for the classification of images, with the use of the pretrevery model for the allocation of symptoms. A comparative analysis of the efficiency of different models was carried out. Conclusions are made on the effectiveness of these models, and recommendations for product development.

CLASSIFICATION, RECOGNITION OF IMAGES, AUTOMATIC DIAGNOSTICS, RECOGNITION OF MEDICAL IMAGES, FUZZY NEURAL NETWORKS

ВСТУП

Оцінка сучасного стану проблеми

У задачах медичної діагностики значною частину проблематики складає виділення признаків для подальшої обробки данних, та вибір методу класифікації признаків з допомогою яких діагностуються хвороби. З розвитком і розповсюдженням Систем підтримки прийняття рішень зростають й вимоги до алгоритмів навчання, та точності. Надійність системи, а також простота використання, впливають на стабільність і швидкість прийняття рішення, що може допомогти швидше назначити лікування, якщо це стосується медицини. Перевага систем штучної діагностики в тому, що вони частково вирішують цю проблему. Методи класифікації і візуалізації дуже гнучкі ,справляются з задачею з невеликима затратами на них. Зараз медицина поступово почала збільшувати використання машинного навчання, на поприщі якого виникають цілі професії. Медична культура і охорону здоров'я зараз на порозі технічної революції. Незважаючи на порівняно молодий вік медичної інформатики, яка налічує не більше ніж 40 років, інформаційні технології стрімко входять в усі сфери медицини і організації охорони здоров'я (сімейна медицина, перехід до страхової медицини, створення єдиного інформаційного простору, інтеграція в європейський медичний простір).

Сьогодні практично жоден етап діагностики не обходиться без комп'ютерних технологій. Поряд з цим, інтелектуальні інформаційні системи досить обмежено застосовують в практичній медицині. Метою інтелектуальних автоматизованих систем є розширення кола завдань, що вирішуються за допомогою комп'ютерів, підвищення рівня інтелектуальної підтримки сучасного лікаря - фахівця, а ключовий завданням використання цих систем є створення методу, що імітує роботу експерта певної сфери. Застосування інтелектуальних систем в медицині, безсумнівно, сприяє прогресивному розвитку інформаційного потенціалу, який є

універсальним засобом вирішення широкого кола завдань в різних етапах лікування пацієнта.

Актуальність

Сьогодні прогнозування на данних що приходять з медичних систем стало реалістичним, воно вимагає все кращих алгоритмів та обчислювальніої потужності. Саме тому дослідники, що стоять на передових наукових позиціях, звертаються до комп'ютерніх систем для отримання максимально можливої обчислювальної потужності. Аналіз складності актуальних обчислювальних задач різних галузей науки і техніки показує, що для їх розв’язку необхідні комп’ютери з дуже високою потужністю.

Рак молочної залози наразі один з найбільших проблем для жіночого здоров'я та смертність від нього одна з самих високих для жінок. Саме тому розвиток алгоритмів для медичної діагностики може допомогти в вирішенні данно ї проблеми.

Гістопатологичний аналіз є надзвичайно важкою та затратною в часі задачею, що залежить від досвиду потологів та впливу факторів таких як усталість та розсіяна увага спеціаліста.

Метою роботи є аналіз та розробка програмних і алгоритмічних засобів підтримки прийняття рішень у медичній галузі дало б можливість вплинути на більш ранні постановки діагнозу також серйозним питанням є використання і впровадження наявних медичних інтелектуальних інформаційних систем.

Медичні інформаційні технології з теоретичної та ексклюзивної для деяких клінік сьогодні впритул наблизилися до буденної медичної практики. Інтелектуальна інформаційна система - це один з видів автоматизованих інформаційних систем, ще Інтелектуальна інформаційну систему називають

системою, заснованою на знаннях.. В процесі моделювання і побудови ІС використовується поняття моделі предметній області, яка на основі системи знань забезпечує автоматичний вибір оптимального алгоритму розв'язання задачі.

У загальному випадку всі системи, засновані на знаннях, можна розділити на системи, вирішують завдання аналізу, і на системи, які вирішують завдання синтезу. Основна відмінність завдань аналізу від завдань синтезу полягає в тому, що якщо в задачах аналізу безлічі рішень може бути перераховане і включене в систему, то в завданнях синтезу безліч рішень потенційно не обмежені і будуються з рішень компонент або проблем. Завданнями аналізу є: інтерпретація даних, діагностика, підтримка ухвалення рішення; до завдань синтезу відносяться проектування, планування, управління. Існують також комбіновані завдання: вивчення, моніторинг, прогнозування Відомі мови взаємодії та подання знань в інтелектуальних системах (Lisp, Prolog, QBE), як правило, мають вузькоспеціалізовану спрямованість. В данній роботі було розглянуто систему для медичної діагностики.

Як відомо, особливістю таких систем є автоматизація вибору і прийняття оптимальних рішень на основі отриманого людиною досвіду і раціонального аналізу зовнішніх впливів, описаних в термінах моделі предметній області. Автоматизація процесів медичної діагностики, як один з найважливіших нап- рямів медицини, відіграє значну роль в підвищенні надійності і точності діагностики захворювань Аналіз наявних автоматизованих систем медичної діагностики показав, що вони не в повній мірі задовольняють вимоги до вирішення завдань, які вимагають складних логічних висновків в умовах високого ступеня невизначеності, неповноти та суперечливості вихідних даних. Вихід із цього становища бачиться в інтелектуалізації цих систем на основі нових інформаційних технологій і, зокрема, в застосуванні концепції експертних систем, які допомагають людині при вирішенні завдань, які важко формалізувати.

1. ОГЛЯД ПРОБЛЕМАТИКИ. ОПИС ТА ВИБІР ВИКОРИСТОВАНИХ СИСТЕМ ТА МАТЕМАТИЧНИХ МЕТОДІВ

1.1 Огляд проблематики

Сьогодні рак - це велика проблема здоров'я в усьому світі. За даними Міжнародного агентства для дослідження раку (IARC), частина Всесвітнього здоров'я Організація (ВООЗ), внаслідок якої було зареєстровано 8,2 млн. смертей рак у 2012 році та 27 мільйонів нових випадків захворювання очікується, відбудеться до 2030 року [1]. Серед типу раку, рак молочної залози є другим найбільш поширеним для жінок. Крім того, смертність від нього дуже висока в порівнянні з іншими видами раку.

Останні досягнення у розумінні молекулярної біології прогресу раку молочної залози та відкриття нових споріднених молекул маркерів, гістопатологічний аналіз залишається найбільшим широко використовуваним методом діагностикі раку молочної залози [2]. Незважаючи на значний прогрес, досягнутий діагностичними технологіями, фінальний діагноз раку молочної залози, включаючи класифікацію та постановку діагнозу, продовжує бути зроблений патологами, що застосовують візуальний огляд гістологічних зразків під мікроскопом. Останні досягнення у технології обробки зображень та машинного навчання дозволяють будувати Системи автоматичного виявлення / діагностики (CAD / CADx) що може допомогти патологам бути більш продуктивними та об'єктивними і послідовними в діагностиці. Класифікація гістопатології зображення на різні шаблони гістопатології, відповідні для неракових або ракових станів аналізу тканини, часто є першочерговою ціллю в системах аналізу зображень для автоматичної діагностики раку. Головною проблемою таких систем є те що вони мають справу з складністю з гістопатологічних образів.

Використовуючи різні моделі машинного навчання, такі як нейронні мережі та SVM , точність результатів становить від 76% до 94% на наборі даних з 92 зображень.

Чжан та ін. [7] пропонують каскадний підхід. На першому рівні каскаду автори відкидують легкі випадки(ті що очевидно не проходять), а інші відправляються на другий рівень, де є більш складна система класифікації використаний метод у запропонованій базі даних ізраїльським технологічним інститутом, до складу якого входять 361 зображення та результати точності 97%. В іншому, ті самі автори оцінюють ансамбль однокласних класифікаторів на тій самій базі даних, що досягає рівня від 92%. Більшість з цих останніх робіт відносяться до класифікації раку молочної залози орієнтовані на цілісне зображення (WSI) [7], [8], [6], [4], [9]. Однак широке впровадження BIC та інших форм з цифрової патології, яка все ще стикається з такими перешкодами, як висока вартість впровадження та експлуатації технології, недостатньо недостатня продуктивність для великих об'ємів клінічних процедур, внутрішні технологічні проблеми, невирішені регуляторні питання, а також «культурний опір» від патологоанатомістів [10]. До недавнього часу більшість робіт на Гістопатологічному аналізі раку молочної залози були проведені на малих наборах даних, які зазвичай недоступні науковому співтовариству. Покращення цього, представив набір даних з 7909 грудей гістопатологічних зображень, отриманих у 82 пацієнтів. У цьому ж дослідженні автори оцінювали різні різноманітні текстурні дескриптори та різні класифікатори, а також повідомляли про серію експериментів з точністю від 80% до 85%, залежно від збільшення чисельності зображень. На підставі результатів, представлених у роботах 12,13 не можна заперечувати, що дескриптори текстур можуть пропонувати хороше уявлення про підготовку класифікаторів. Проте деякі дослідники виступають за те, що основна слабкість сучасних методів машинного навчання виявляється саме на цій особливості технічного кроку. Все це означає, що алгоритми машинного навчання повинні бути менш залежними від функціональної інженерії, будучи здатними витягати та організовувати дискримінаційну інформацію з даних, іншими словами, повинні

бути здатними вивчати представлення. Ідея представлення навчання не нова, але вона з'явилася лише недавно як життєздатна альтернатива у зв'язку з появою тампопулярністю блоків графічної обробки (графічних процесорів), які здатні забезпечити високу обчислювальну пропускну здатність за відносно невелику вартість, досягнуті завдяки їх масово паралельній архітектурі.

Окрім того в данній роботі ми розглядаємо підхід використання попередньо навченої моделі,(що навчалась на зовсім інших даних).

Крім різних підходів, ConvolutionalNeural Network (CNN), представлена Лекуном в [14], широко використовується для досягнення найновіших результатів при різних проблемах розпізнавання, образів мікроскопічної та макроскопічної текстури. Показано, що CNN здатний перевершувати традиційні текстурні дескриптори. Крім того, традиційний підхід до виявлення відповідних ознак для класифікації у патологічних зображеннях вимагає значних зусиль та ефективного знання експертного середовища, що часто призводить до високоінтегрованих рішень, специфічних для кожної задачі і навряд чи застосовних в інших контекстах. З огляду на це, в цій роботі ми оцінюємо глибокий підхід до вивчення проблеми гістопатологічної класифікації зображень Раку молочної залози. Та досліджуємо можливість використання вже навчених моделей

Набір експериментів із набору даних BreaKHis, запропонований в [11], показує, що CNN досягає кращих результатів, ніж найкращі результати, отримані іншими моделями машинного навчання, які пройшли навчання за допомогою текстурних сценаріїв. Найкраща продуктивність, однак, отримується шляхом поєднання різних CNN

1.2 Опис даних

База даних BreaKHis [11] містить мікроскопічні біопсії з доброякісних та злоякісних пухлин молочної залози. Зображення були відібрані в клінічному дослідженні з січня 2014 року по грудень 2014 року. BreaKHis складається з 7909 клінічно репрезентативних мікроскопічних зображень грудних опухолей зібраних у 82 пацієнтів з різними збільшеннями (40X, 100X, 200X, 400X).

Всі пацієнти, на протязі цього періоду обслідувались в лабораторії P&D, Бразилія, з клінічними показаннями раку молочної залозі, були запрошені взяти участь у дослідженні. Інституційна ревізійна комісія схвалила дослідження і всі пацієнти дали письмову інформовану згоду. Всі дані були анонімізовані.

Зразки генеруються з біопсійних слайдів грудної клітки, забарвлених гематоксиліном та еозином (HE). Зразки збираються хірургічною (відкритою) біопсією (СОБ), підготовленою для гістологічного дослідження і поміченої патологами лабораторії P&D. Процедура підготовки, що використовується в цій роботі, є стандартним процесом, який широко використовується в клінічній практиці. Основна ціль полягає у збереженні оригінальної структури тканини та молекулярної композиції, що дозволяє спостерігати її за допомогою світлового мікроскопа. Повна процедура підготовки включає етапи, такі як фіксація, зневоднення, очищення, інфільтрація, введення та обробки. Для монтування на слайди, секції довжиною близько 3мкм розрізаються за допомогою мікротома. Після фарбування секції накриваються скляним покривним склом. Тоді патологи ідентифікують пухлинні ділянки в кожному слайді, шляхом візуального аналізу тканинних ділянок під мікроскопом. Остаточний діагноз кожного випадку виробляють досвідчені патологи і підтверджують додатковими екзаменами, такими як аналіз імуногістохімії (IHC). Система мікроскопа Olympus BX-50 з релейним об'єктивом зі збільшенням 3,3 ×, з'єднаною з цифровою камерою Samsung SCC-131AN, використовується для отримання оцифрованого зображення з тканин молочної залози. Зображення отримані в 3-канальному колірному просторі

TrueColor (24-розрядна глибина кольорів, 8-бітовий кольоровий канал) RGB (Red-Green-Blue) з коефіцієнтами збільшення 40 ×, 100 ×, 200 × і 400 × відповідно до об'єктиву 4 ×, 10 ×, 20 × і 40 ×. На малюнку показані чотири зображення - з чотирма збільшувальнимифакторами 40 ×, 100 ×, 200 × і 400 × - отриманими з одного слайду тканини молочної залози, що містить злоякісну пухлину (рак молочної залози). Виділений прямокутник (вручну доданий лише для ілюстративних цілей) - це область інтересів, вибрана патологом, яка буде деталізована у наступному збільшенні. На сьогодні база даних складається з 7909 зображень, поділених на доброякісні та злоякісні пухлини. Таблиця 1.1 підсумовує розподіл зображення.

Таблиця 1.1 - РОЗПОДІЛ ЗОБРАЖЕННЯ ФАКТОРОМ ЗБІЛЬШЕННЯ ТА КЛАСУ

Збільшення	Доброякісні	Злоякісні	Всього
40x	625	1370	1995
100x	644	1437	2081
200x	623	1390	2013
400x	588	1232	1820
Всього	2480	5429	7909
Пацієнти	24	58	82

Також в данному наборі данних зображення розділені по типу пухлин далі в рисунках буде наведено різні збільшення одного й того самого рисунку (див рис 1.1 - 1.4)

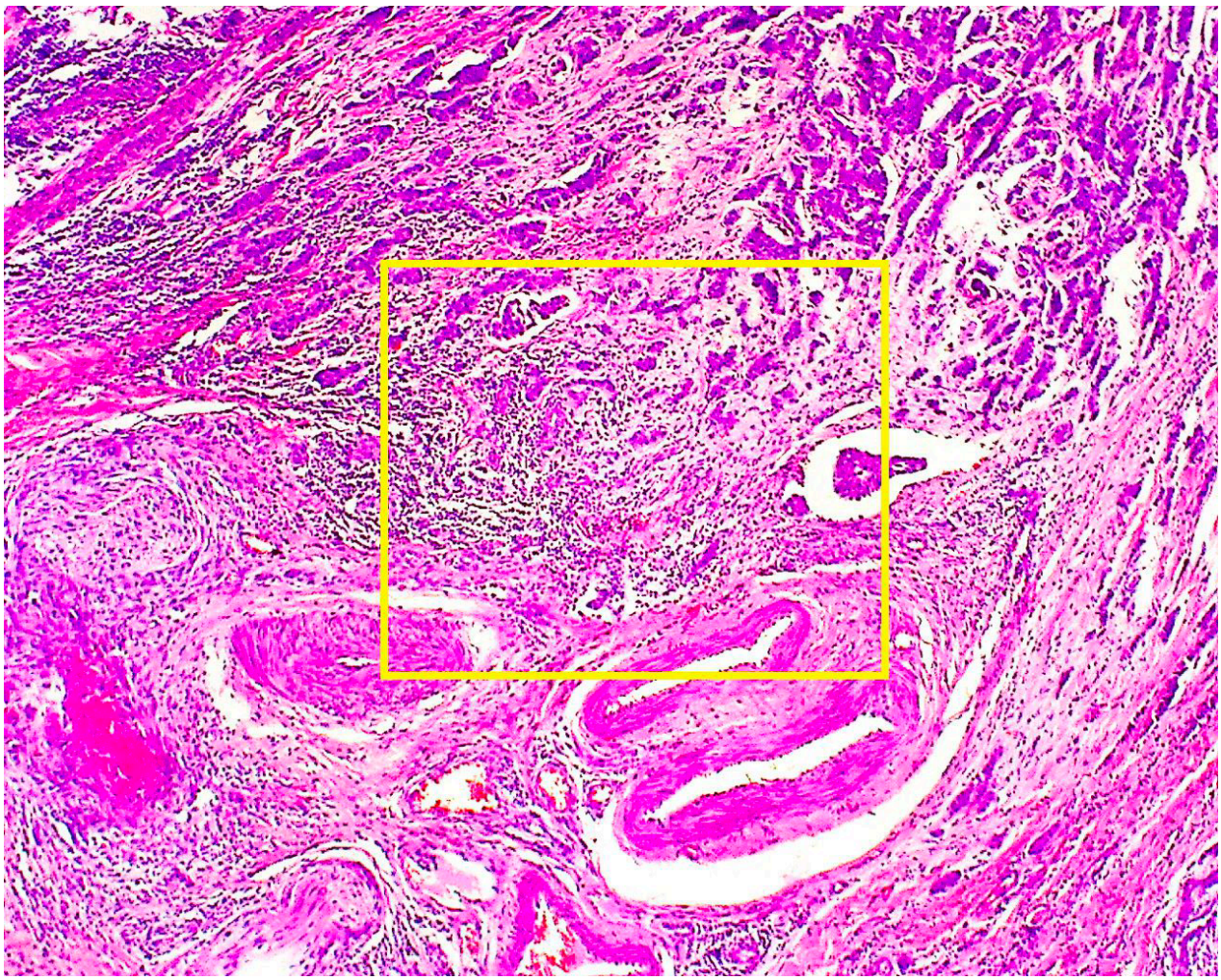


Рисунок 1.1 - Слайд злоякісної пухлини в збільшенні 40X

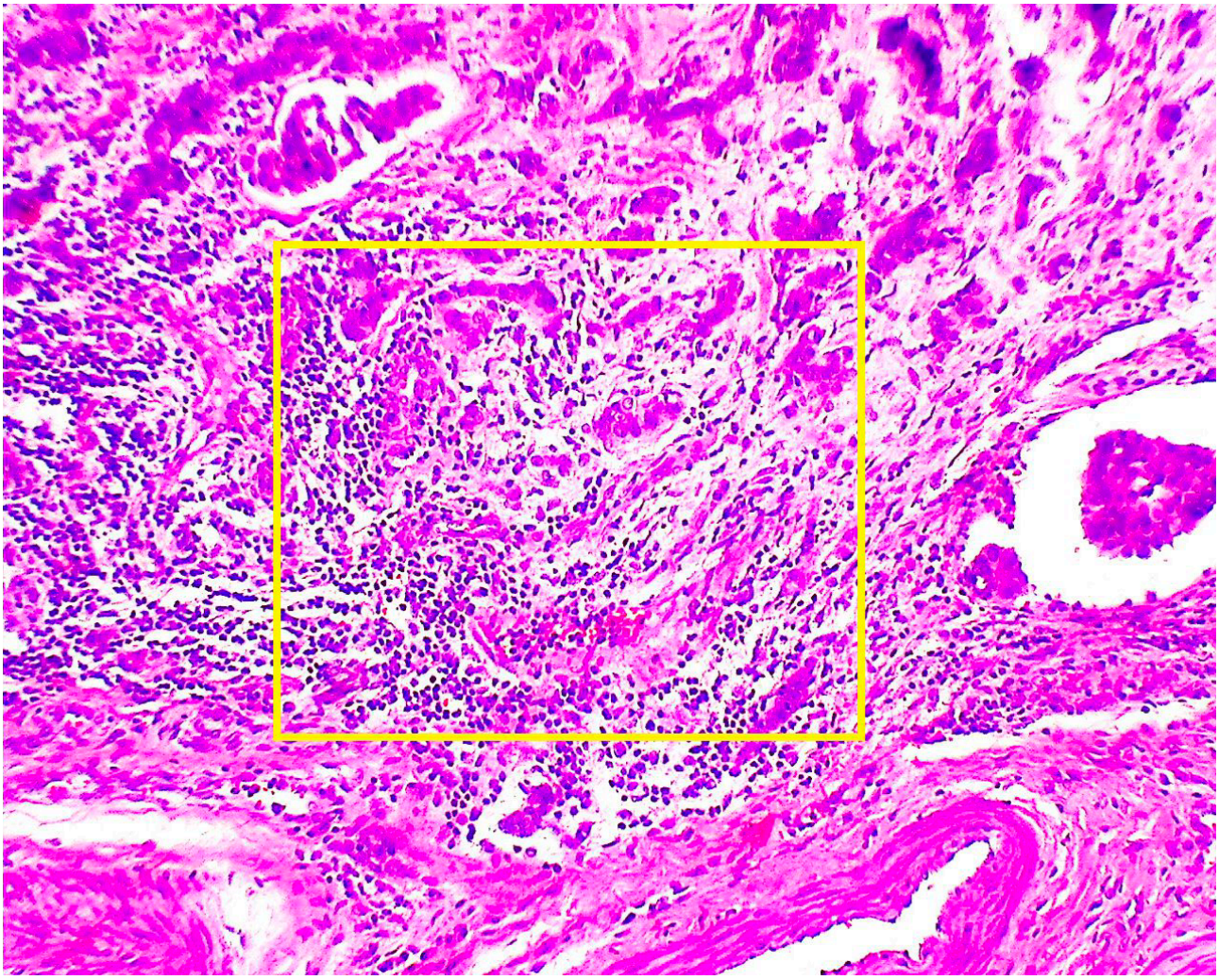


Рисунок 1.2 - Слайд злоякісної пухлини в збільшенні 100X

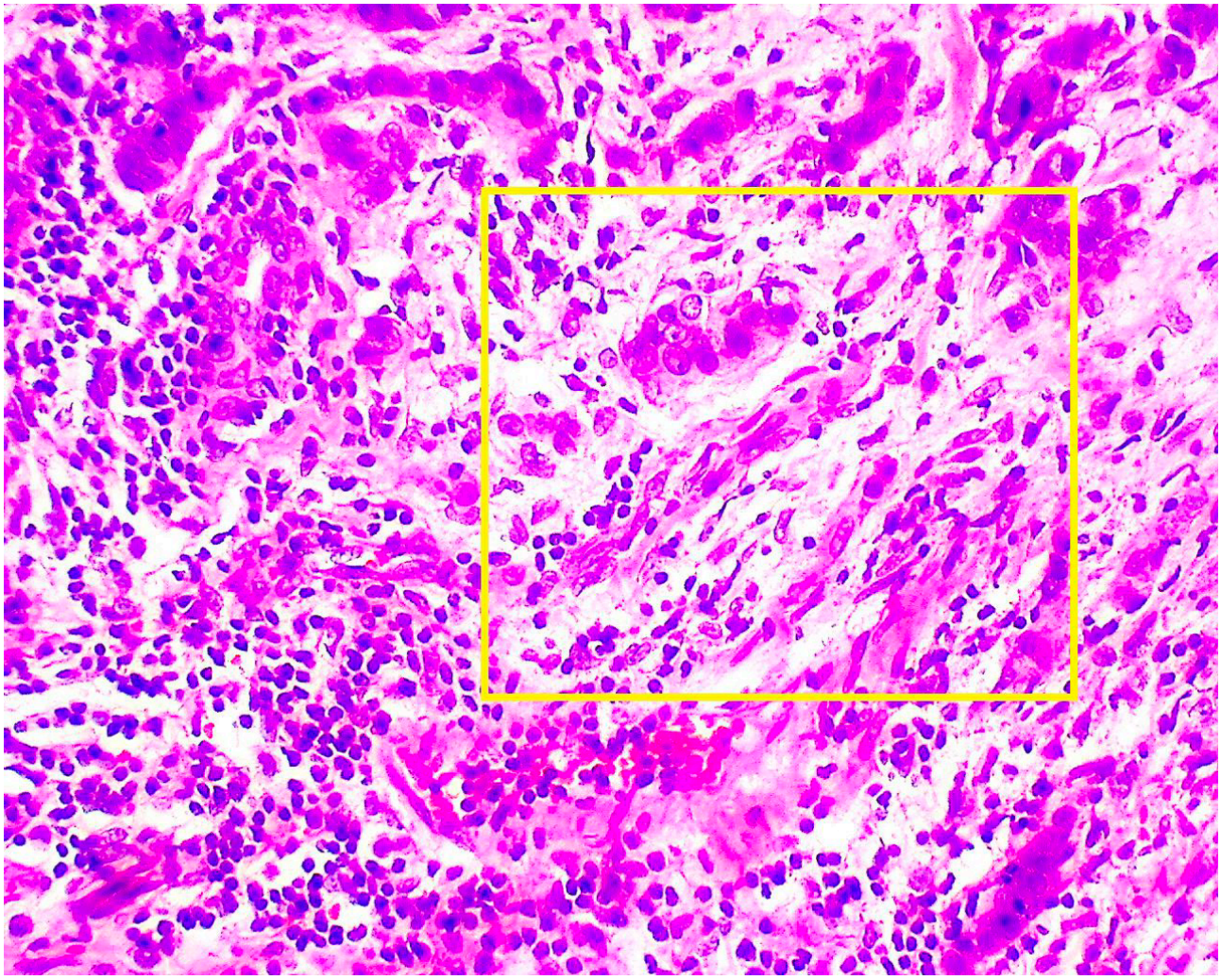


Рисунок 1.3 - Слайд злоякісної пухлини в збільшенні 200X

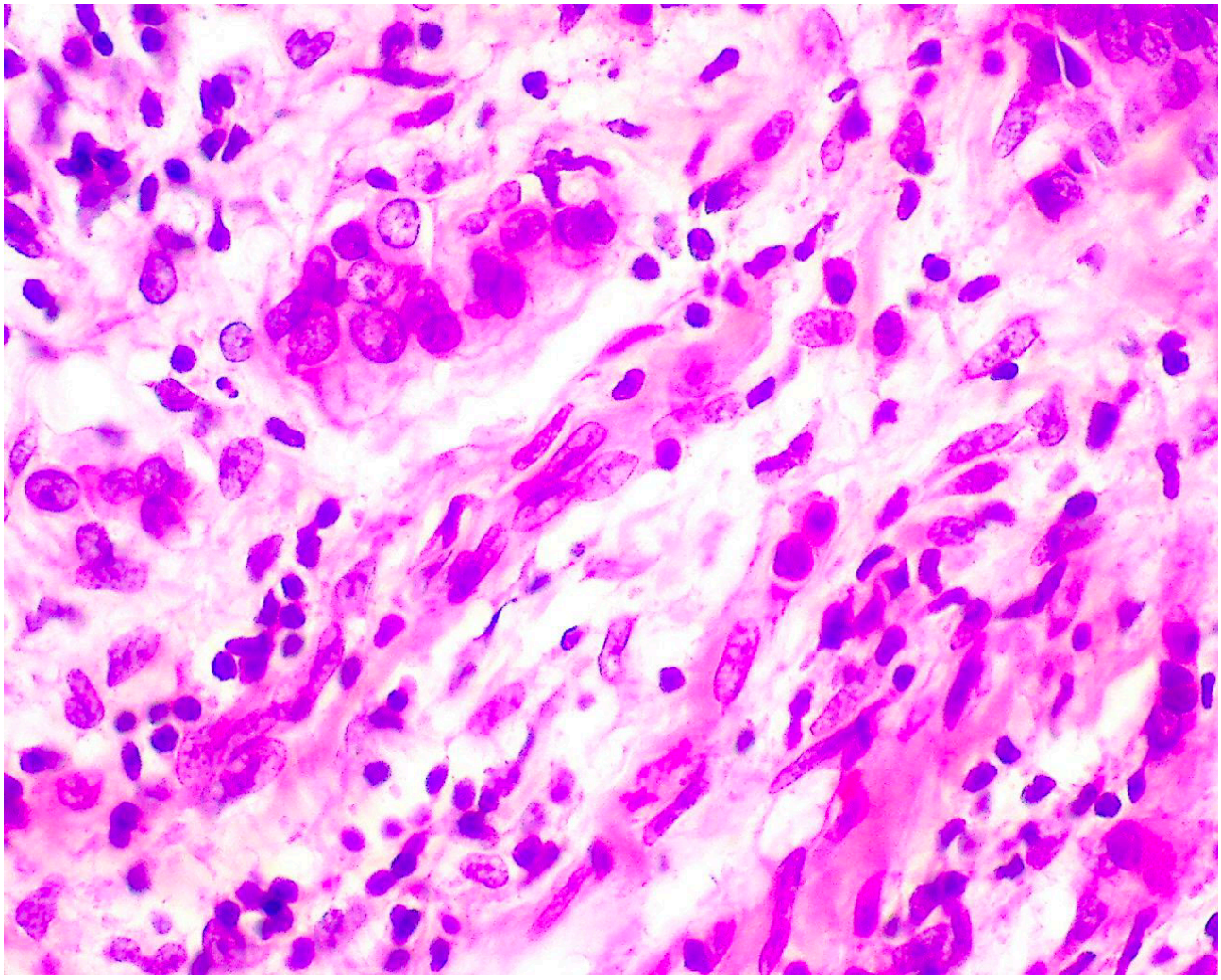


Рисунок 1.4 - Слайд злоякісної пухлини в збільшенні 400X

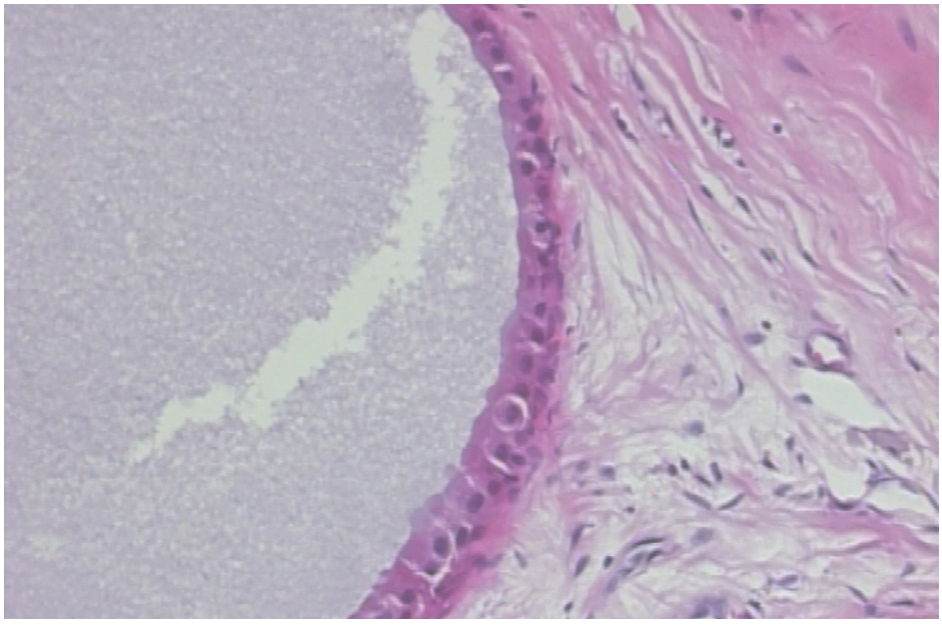


Рисунок 1.5 – Слайд доброякісної пухлини в збільшенні 100X

1.3 Опис моделі NEFCLASS

Метою моделі NEFCLASS (NEuro Fuzzy CLASSifier) є отримання нечітких правил з безлічі даних, які можна розділити на різні класи. Нечіткі правила описують дані в формі:

R: якщо $\varepsilon_j \in x_j$. має функцію приналежності $\mu_1,\ x_2 - \mu_2, \ldots x_n - \mu_{nj}.$,

то зразок належить класу і, де - нечіткі множини. Завдання NEFCLASS полягає в тому, щоб визначити приналежність до класу вхідного зразка. Мається на увазі, що перетин двох різних множин порожньо. Розглянемо більш детально архітектуру моделі NEFCLASS.

База правил являє собою апроксимацію невідомої функції і описує класифікаційну задачу, де така, що і x належить класу.

Нечіткі множини та лінгвістичні правила представляють апроксимацію і визначають результат системи NEFCLASS. Вони виходять з безлічі вибірок шляхом навчання. Обов'язково повинно виконуватися правило, що для кожного лінгвістичного значення може існувати тільки одне подання нечіткої множини.

Система NEFCLASS має 3-шарову послідовну архітектуру. Перший шар містить вхідні нейрони, в яких представляються вхідні зразки. Активація нейрона зазвичай не змінює вхідного значення. Прихований шар містить нечіткі правила, і третій шар складається з вихідних нейронів кожного класу. Активація для нейронів правил і для нейронів вихідного шару з зразком p обчислюється так:

$$a_R^{(p)} = \min_{x \epsilon U_1} \left\{ W(x,R)(a_x^{(p)}) \right\}$$

$$a_R^{(p)} = \sum_{R \epsilon U_2} W(c,R)(a_R^{(p)})$$

де W (x, R) - нечітка вага з'єднання вхідного нейрона x з нейроном правила R, а W (R, c) - нечіткий вага з'єднання нейрона правила R з нейроном вихідного шару c. Замість застосування операцій взяття максимуму і мінімуму можна використовувати інші функції t-норми і t-конорми відповідно.

База правил являє собою апроксимацію невідомої функції і описує класифікаційну задачу, де така, що і x належить класу.

Нечіткі множини та лінгвістичні правила представляють апроксимацію і визначають результат системи NEFCLASS (див рис 1.5). Вони виходять з безлічі вибірок шляхом навчання. Обов'язково повинно виконуватися правило, що для кожного лінгвістичного значення може існувати тільки одне подання нечіткої множини.

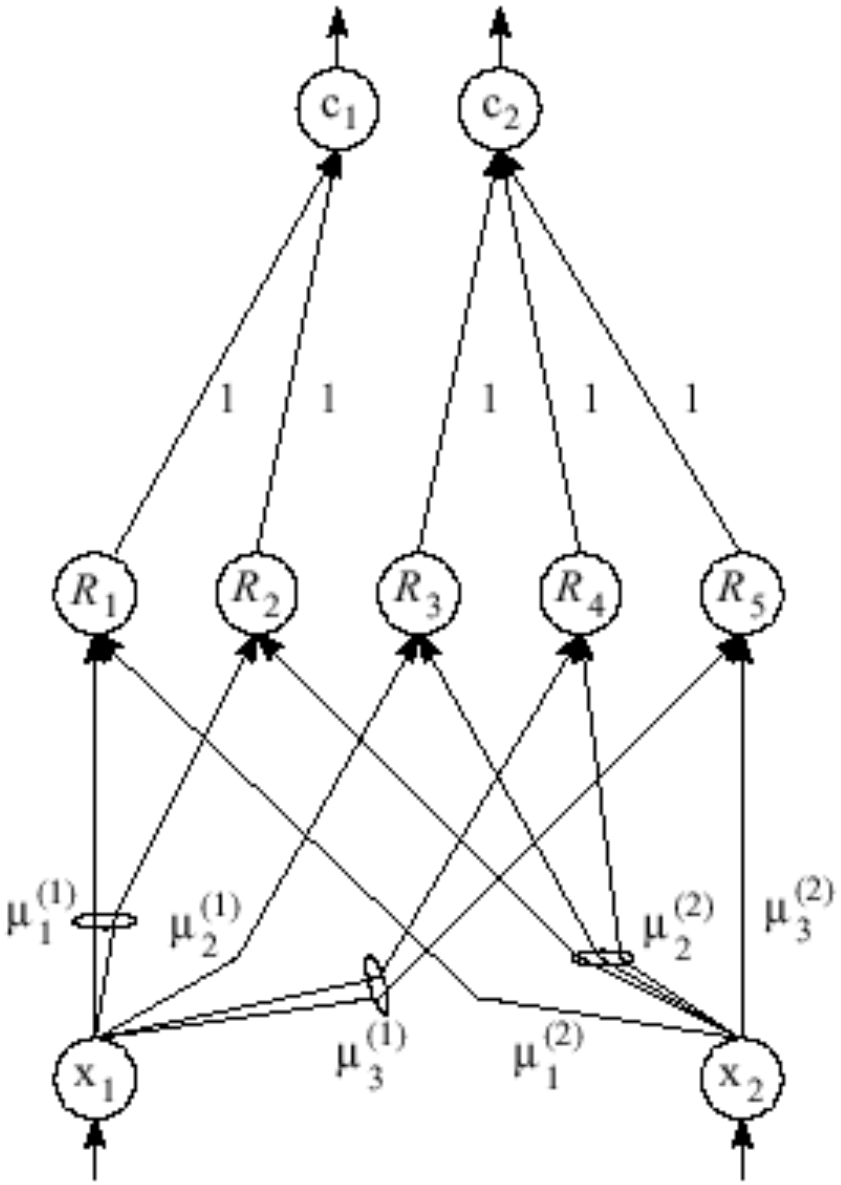


Рисунок 1.6 - Схема системи NEFCLASS

1.4 Опис методу найкоротшого спуску

Всі методи спуска рішення задачі безумовної мінімізації відмінностей-ються або вибором напрямку спуска, або способом руху вздовж напрямку спуску. Це дозволяє написати загальну схему методів спуска.

Вирішується завдання мінімізації функції (x) на всьому просторі En. Методи спуску складаються в наступній процедурі побудови послідовно-сті {xk}. В якості початкового наближення вибирається будь-яка точка x0 En. Послідовні наближення x1, x2, ... будуються за такою схемою:

а) в точці xk вибирають напрям спуску - Sk;

б) знаходять (k + 1) -е наближення за формулою xk + 1 = xk-hkSk.

Напрямок Sk вибирають таким чином, щоб забезпечити нерівність f (xk + 1) <f (xk) принаймні для малих значень величини hk. На питання, ка-кому із способів вибору напрямку спуска варто віддати перевагу при вирішенні конкретної задачі, однозначної відповіді немає.

Число hk визначає відстань від точки xk до точки xk + 1. Це число називається довжиною кроку або просто кроком. Основне завдання при виборі ве-личини hk - це забезпечити виконання нерівності (xk + 1) <(xk).

Величина кроку сильно впливає на ефективність методу. Більшої ефек-тивностью володіє варіант методу, коли крок за кожною змінною визна-ляется напрямними косинусами градієнта (в градієнтних методах).

$$x_{k+1}=x_k-h_k \cos\phi_i$$

де - $\cos\phi_i=\frac{(dR/dx_j)}{|gradR(x)|}$

У цьому випадки величина робочого кроку не залежить від величини модуля градієнта, і нею легше керувати зміною h. В районі оптимуму може виникати значне "нишпорення", тому використовують різні алгоритми-ми корекції h.

Найбільшого поширення набули такі алгоритми:

1. $h^i = const = h$ (без коррекции);

2. $h^i = h^{i-1}/2$ если $R(x^i) < R(x^{i-1})$; $h^i = h^{i-1}$ если $R(x^i) > R(x^{i-1})$

3. $h^i = h^{i-1}$, если $\alpha_1 \le \alpha \le \alpha_2$; $h^i = 2h^{i-1}$, если $\alpha_1 > \alpha$; $h^i = \frac{h^{i-1}}{3}$, если $\alpha_2 < \alpha$,

где α –кут між градієнтами на попередньому и теперішньому кроці;

α_1 и α_2 – заданные пороговые значення вибираются субєктивно

(например, $\alpha_1 = \pi/6, \alpha_2 = \pi/3$).

Далеко від оптимуму напрямок градієнта змінюється мало, тому крок можна збільшити (другий вираз), поблизу від оптимуму напрямок різко змінюється (кут між градієнтами R (x) великий), тому h скорочується (третє вираз).

Метод градієнтного спуску.

Розглянемо функцію f, вважаючи для визначеності, що вона залежить від трьох змінних x, y, z. Обчислимо її приватні похідні df / dx, df / dy, df / dz і образовуюче з їх допомогою вектор, який називають градієнтом функції:

grad f(x, *y, z*) = *∂f (x, y,z) /∂x*****i***+*∂f(x, y, z)/∂y*****j***+*∂f(x, y,z)/∂z*****k***.

Тут i, j, k - одиничні вектори, паралельні координатним осях. Приватні похідні характеризують зміна функції f по кожній незалежної змінної окремо. Утворений з їх допомогою вектор градієнта так-ет загальне уявлення про поведінку функції в околі точки (x, y, z). Напрямок цього вектора є напрямом найбільш швидкого возрас-вання функції в даній точці. Протилежне йому напрямок, яке часто називають антіградієнтним, являє собою напрям найбільш швидкого спадання функції. модуль градієнта

$$|\text{grad } (x, y,z)| = \sqrt{(\partial f/\partial x\ (x, y,z))^2 + (\partial f/\partial y(x, y, z))^2 + (\partial f/\partial z(x, y,z))^2}.$$

визначає швидкість зростання і спадання функції в напрямку гради-ента і антіградієнта. Для всіх інших напрямків швидкість зміни функції в точці (x, y, z)

менше модуля градієнта. При переході від однієї точки до іншої як напрямок градієнта, так і його модуль, взагалі кажучи, міняються. Поняття градієнта природним чином переноситься на функції будь-якого числа змінних.

Перейдемо до опису методу градієнтного спуску. Основна його ідея полягає в тому, щоб рухатися до мінімуму в напрямку найбільш швидкого зменш-ня функції, яке визначається антіградієнтом. Ця ідея реалізується в такий спосіб.

Виберемо будь-яким способом початкову точку, обчислимо в ній градієнт даної функції і зробимо невеликий крок в зворотному, Антиград-ентном напрямку. В результаті ми прийдемо в точку, в якій значення функції буде менше початкового. У новій точці повторимо процедуру: знову обчислимо градієнт функції і зробимо крок в зворотному напрямку. Продовжуючи цей процес, ми будемо рухатися в сторону зменшення функції. Спеціальний вибір напрямку руху на кожному кроці дозволяє Наді-яться на те, що в даному випадку наближення до найменшого значення функції буде швидшим, ніж в методі покоординатного спуску.

Метод градієнтного спуску вимагає обчислення градієнта цільової функ-ції на кожному кроці. Якщо вона задана аналітично, то це, як правило, не проблема: для приватних похідних, що визначають градієнт, можна напів-чить явні формули. В іншому випадку приватні похідні в потрібних точках доводиться обчислювати наближено.

Для оцінки приватних похідних використовуються різницеві методи:

1.Алгорітм з центральної пробою

$$\frac{dR}{\Delta x_i} \approx \frac{f(x_1, \ldots, x_i + g_i, \ldots, x_n) - f(x_1, \ldots, x_i, \ldots, x_n)}{g_i}$$

2. Алгоритм з парними пробами

$$\frac{dR}{\Delta x_i} \approx \frac{f(x_1, \ldots, x_i + g_i, \ldots, x_n) - f(x_1, \ldots, x_i - g_i, \ldots, x_n)}{2g}$$

де gi - пробний крок по і-й змінної, обираний досить малим для різницевої оцінки похідної.

Відзначимо, що при таких розрахунках gi, можна брати занадто малим, а значення функції потрібно обчислювати з досить високим ступенем точності, інакше при обчисленні різниці

$$\Delta f(x_1, ...,x_i+ g_i, ..., x_n) - f(x_1, ..., x_i, ..., x_n)$$
$$\Delta f(x_1, ...,x_i+ g_i, ..., x_n) - f(x_1, ..., x_i- g_i,..., x_n)$$

буде допущена велика помилка.

Перший алгоритм вимагає менших витрат в порівнянні з другим (зазвичай витрати виражаються кількістю обчислень критерію оптимальності), але дозволяє отримати рішення менш точно, ніж другий, ця похибка зави-сит від величини пробного кроку

На рис 1.6 зображені лінії рівня функції двох змінних u = f (x, y),, і приведена траєкторія пошуку її мінімуму за допомогою методу градієнтного спуску.

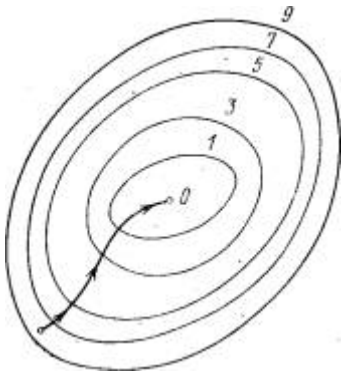


Рисунок 1.7 - Ілюстрація спуску

Метод покоординатного спуску.

Нехай потрібно знайти найменше значення цільової функції

u = f (M) = f (x, x,..., xn). Тут через M позначена точка n-мірного простору-ства з координатами x, x,. . . , Xn: M = (x, x,..., Xn). Виберемо яку-небудь початкову точку M = (x, x,..., Xn0) і розглянемо функцію f при фіксовані-ванних значеннях всіх змінних, крім першої: f (x, x, x,..., Xn0). Те гда вона перетвориться в функцію однієї змінної x. Змінюючи цю змін-ву, будемо рухатися від початкової точки x = x в сторону зменшення функ-ції, поки не дійдемо до її мінімуму при x = x, після якого вона починає зростати. Крапку з координатами (x, x, x,..., Xn0) позначимо через M, при цьому f (M0) f (M).

Фіксуємо тепер змінні: x = x, x = x,. . . , Xn = xn0 і розглянемо функцію f як функцію однієї змінної x: f (x, x, x..., Xn0). Изме-ня x, будемо знову рухатися від початкового значення x2 = x20 в сторону уби-вання функції, поки не дійдемо до мінімуму при x2 = x21 .Точку з Координа-тами

{X, x, x. . . xn0} позначимо через M, при цьому f (M1) f (M).

Проведемо таку ж мінімізацію цільової функції по змінним

x, x,. . . , Xn. Дійшовши до змінної xn, знову повернемося до x і продовжимо процес. Ця процедура цілком виправдовує назву методу. З її допомогою ми побудуємо послідовність точок M, M, M,. . . , Якій відповідає монотонна послідовність значень функції

f (M0) f (M) f (M) Обриваючи її на певному етапі k можна прибл-женно прийняти значення функції f (Mk) за її найменше значення в рас-сматрівать області.

Проведемо таку ж мінімізацію цільової функції по змінним x, x,. . . , Xn. Дійшовши до змінної xn, знову повернемося до x і продовжимо про-процес. Ця процедура цілком виправдовує назву методу. З її допомогою ми побудуємо

послідовність точок M, M, M,. . . , Якій відповідає монотонна послідовність значень функції

f (M0)>= f (M1)>= f (M2) Обриваючи її на певному етапі k можна наближено прийняти значення функції f (Mk) за її найменше значення в даній області.

Відзначимо, що даний метод зводить задачу пошуку найменшого значення функції кількох змінних до багаторазового вирішення одновимірних задач оптимізації. Якщо цільова функція f (x, x, ..., xn) задана явною формулою і є дифференцируемой, то ми можемо ви-числиться її приватні похідні і використовувати їх для визначення направ-лення спадання функції по кожній змінній і пошуку відповідних одновимірних мінімумів. В іншому випадку, коли явною формули для це-лівої функції немає, одномірні завдання слід вирішувати за допомогою одномірних методів

На рис. зображені лінії рівня деякою функції двох змінних u = f (x, y). Уздовж цих ліній функція зберігає постійні значення, що дорівнюють 1, 3, 5, 7, 9. Показана траєкторія пошуку її найменшого зна-чення, яке досягається в точці O, за допомогою методу покоординатного спуску. При цьому потрібно чітко розуміти, що малюнок є тільки для иллю-страції методу.

Нехай потрібно вирішити задачу (2):

f (x) min, x Rn. (2)

У двовимірному просторі R2. Рішення завдання (2) методом покоординат-ного спуску, інакше званого методом Гаусса - Зейделя, виробляють за такою загальною схемою.

Вибирають довільно початкову точку x (0) з області визначення функції f (x). Наближення x (k) визначаються співвідношеннями

(3): x (k + 1) = x (k) + t (k) S (k) (k = 0,1,2, ...),

де вектор напрямку спуску s (k) - це одиничний вектор, що збігається з яким-небудь координатним напрямком (наприклад, якщо S (k) паралельний x1, то S (k)

= {1,0,0, ..., 0} , якщо він паралельний x2, то S (k) = {0, 1, 0,..., 0} і т.д.); величина t (k) є рішенням задачі одновимірної мінімізації:

f (x (k) + ts (k)) min, t R1, (k = 0,1,2, ...),

і може визначатися, зокрема, методом сканування (див рис 1.8)

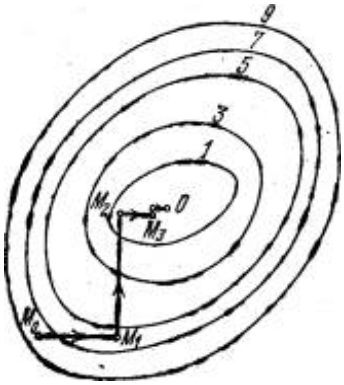


Рисунок 1.8 - Іллюстрація методу покоординатного спуску

Детальна реалізація загальної схеми в двовимірному випадку R2 дає траєкторій наближення до точки x * методом покоордінатного спуску, що складається з зве-Ньєво ламаної, що з'єднують точки x (k), x1 (k) x (k + 1) (k = 0, 1, 2,). При k = 0, виходячи з початкової точки x (0) = (x1 (0), x2 (0)), знаходять точку x (0) = (x1 (0), x2 (0)), мінімуму функції однієї змінної f (x1, x2 (0)); при цьому f (x (0)) f (x (0)). Потім знаходять точку мінімуму x (1) функції f (x1 (0), x2) по другій координаті. Далі роблять наступний крок обчислень при k = 1. Як і лага, що вихідною точкою розрахунку є x (1). Фіксуючи другу координує ДИНАТ точки x (1), знаходять точку мінімуму x (1) = (x1 (1), x2 (1)), функції f (x1, x2 (1)) однієї змінної x (1); при цьому f (x (1)) f (x (1)) f (x (0)). Точку x (2) отримують, мінімізуючи цільову функцію f (x1 (1), x2), знову по коорданате x2, фіксуючи координату x1 (1), точки x (1), і т.д.

Метод найшвидшого спуску

Суть методу найшвидшого спуску полягає в наступному. Як і раніше, в початковій точці визначається антіградієнта мінімізується. Од-нако тепер в напрямку антіградієнта робиться жоден крок, а рухаються в цьому напрямку до тих пір, поки цільова функція спадає, досягає в деякій точці мінімуму. У цій точці знову визначають антіградієнта і шукають нову точку мінімуму цільової функції і так далі. В даному методі спуск має більш цілеспрямований характер, проводиться більшими кроками і градієнт функції обчислюється в меншій кількості точок.

1.5 Дифференціальна еволюція

Диференціальна еволюція (англ. Differential evolution) - метод багатовимірної математичної оптимізації, що відноситься до класу стохастичних алгоритмів оптимізації (тобто працює з використанням випадкових чисел) і використовує деякі ідеї генетичних алгоритмів. Це прямий метод оптимізації, тобто він вимагає тільки можливості обчислювати значення цільової функцій, але не її похідних. Метод диференціальної еволюції призначений для знаходження глобального мінімуму (або максимуму) недиференційовних, нелінійних, мультимодальних (мають, можливо, велика кількість локальних екстремумів) функцій від багатьох змінних. Метод простий в реалізації і використанні (містить мало керуючих параметрів, що вимагають підбору), легко распараллеливается. Метод диференціальної еволюції був розроблений Рейнер сторно і Кеннетом Прайсом, вперше опублікований ними в 1995 році [1] і розвинений в подальшому в їх більш пізніх роботах. [2] [3] Алгоритм В його базовому вигляді алгоритм можна описати таким чином. Спочатку генерується деякий безліч векторів, званих поколінням. Під векторами розуміються точки n-мірного простору, в якому визначена цільова функція f (x),

яку потрібно мінімізувати. На кожній ітерації алгоритм генерує нове покоління векторів, випадковим чином комбінуючи вектори з попереднього покоління. Число векторів в кожному поколінні один і той же і є одним з параметрів методу. Нове покоління векторів генерується в такий спосіб. Для кожного вектора x_i зі старого покоління вибираються три різних випадкових вектора v_1, v_2, v_3 серед векторів старого покоління, за винятком самого вектора x_i, і генерується так званий мутантний вектор (mutant vector) за формулою: v = v_1 + F \ cdot (v_2 - v_3), де F - один з параметрів методу, деяка позитивна дійсна константа в інтервалі [0, 2]. Над мутантним вектором v виконується операція «схрещування» (crossover), яка полягає в тому, що деякі його координати заміщуються відповідними координатами з вихідного вектора x_i (кожна координата заміщається з певною ймовірністю, яка також є ще одним з параметрів цього методу). Отриманий після схрещування вектор називається пробним вектором (trial vector). Якщо він виявляється краще вектора x_i (тобто значення цільової функції стало менше), то в новому поколінні вектор x_i замінюється на пробний вектор, а в іншому випадку - залишається x_i. Приклади практичних додатків Пошукова система Яндекс використовує метод диференціальної еволюції для поліпшення своїх алгоритмів ранжирування.

Основний варіант алгоритму Дифференціальної еволюції полягає в тому, що він має популяцію кандидатів на рішення Ці агенти рухаються навколо в пошуковому просторі, використовуючи прості математичні формули для об'єднання позицій існуючих агентів із популяції. Якщо нова позиція агента є покращенням, вона приймається за основну і становить частину населення, інакше нова позиція просто відкидається. Процес повторюється, і, таким чином, очікується, що врешті-решт знайдеться задовільне рішення.

1.6 Алгоритм “басейного стрибку” (basinhopping)

Це стохастичний алгоритм, який намагається знайти глобальний мінімум гладкої скалярної функції одного або декількох змінних. Алгоритм являється однією з варіацій методу монте карло(алгоритму імітації відпалу).

Алгоритм у його нинішній формі був описаний Девідом Уалесом та Джонатаном Дойє[41]. Це двофазний метод, який поєднує в собі глобальний покроковоий алгоритм з локальною мінімізацією на кожному кроці. Призначений для імітації природних процесів мінімізації енергії кластерів атомів, він добре працює для подібних завдань з "воронко-подібними " енергетичними ландшафтами

Алгоритм ітеративний для кожного циклу, що складається з наступних кроків:

- Задання початкового значення
- Локальна мінімізація
- випадкове спотворення координат
- локальна мінімізація
- прийняти або відхилити нові координати на основі мінімізованого значення функції

Більш повний опис алгоритму можна проглянути в 40

Стохастична частина полягає в тому, щоб виконувати випадкові зміщення або збурення в певній точці - це місцевий мінімум, і це явне зміщення має бути "достатньо великим", щоб уникнути поточного місцевого мінімуму, але не надто великим, щоб запобігти тому що стрибок стає "абсолютно випадковим". Мотивація з фізичної хімії полягає в тому, що стабільні конфігурації - локальні мінімуми - розташовані в тому ж регіоні пошукового простору, так що алгоритм не повинен вивчати весь простір пошуку, що є однією з основних проблем глобальної

оптимізації в загальному випадку Але якщо мінімум "упаковується" в деяку область простору, «скачок басейну» може бути дуже потужним.

Цей глобальний метод мінімізації виявився надзвичайно ефективним для широкого кола проблем фізики та хімії. Це особливо корисно, коли функція має багато мінімумів, розділених великими діапазонами.

Для стохастичної глобальної оптимізації неможливо визначити, чи дійсно був виявлений справжній глобальний мінімум. Замість цього, як перевірка послідовності, алгоритм може бути запущений з ряду різних випадкових відправних точок, щоб забезпечити найнижчий мінімум, знайдене в кожному прикладі, збігом до глобального мінімуму. З цієї причини basin-hopping буде за замовчуванням просто запускати за кількістю ітерацій niter і поверненайнижчий мінімум. Залишається користувачеві гарантувати, що це фактично є глобальним мінімумом.

1.7 Python

Для розробки взаємодії ріалтайм системи з користувачем було розроблено вебсервер що отримує данні з системи та через взаємодію з веб інтерфойсом виводить результат користувачеві.

Python - інтерпретована об'єктно-орієнтована мова програмування високого рівня з динамічною семантикою. Розроблена в 1990 році Гвідо ван Россумом. Структури даних високого рівня разом з динамічною семантикою і динамічним зв'язуванням роблять її привабливою для швидкої розробки програм, а також як засіб поєднання існуючих компонентів. Python підтримує модулі та пакети модулів, що сприяє модульності та повторному використанню коду. Інтерпретатор Python і стандартні бібліотеки доступні як в скомпільованій так і у вихідній формі на всіх

основних платформах. У мові програмування Python підтримується кілька парадигм програмування, зокрема:

об'єктно-орієнтована, процедурна, функціональна і аспектно-орієнтований.

Python транслятори доступні для установки на багатьох операційних системах, дозволяючи виконання коду Python на різноманітних системах.

Використання сторонніх інструментів, таких як py2exe або Pyinstaller, дозволяє Python коду бути упакованим в автономних виконуваних програмах для деяких з найбільш популярних операційних систем, що дозволяє для розповсюдженню програмного забезпечення на основі Python для використання на цих середовищах без установки інтерпретатора Python.

Серед основних її переваг можна назвати наступні:

- чистий синтаксис (для виділення блоків слід використовувати відступи);
- переносимість програм (що властиво більшості інтерпретованих мов);
- стандартний дистрибутив має велику кількість корисних модулів
- (включаючи модулем для розробки графічного інтерфейсу);
- можливість використання Python в діалоговому режимі (дуже корисно
- для експериментування та рішення простих задач);
- стандартний дистрибутив має просте, але разом з тим досить потужне
- середовище розробки, яка називається IDLE і яке написано на мові Python;
- зручний для вирішення математичних проблем (має засоби роботи з комплексними числами, може оперувати з цілими числами довільної величини, в діалоговому режимі може використовуватися як потужний калькулятор).

Python має ефективні структури даних високого рівня і простий, але ефективний підхід до об'єктно-орієнтованого програмування. Елегантний синтаксис Python, динамічна обробка типів, а також те, що це інтерпретована мова, роблять її ідеальною для написання скриптів та швидкої розробки додатків в багатьох галузях на більшості платформ.

Інтерпретатор мови Python і багата стандартна бібліотека (як початкові тексти, так і бінарні дистрибутиви для всіх основних операційних систем)

35

можуть бути отримані з сайту Python www.python.org, і можуть вільно поширюватися. Цей же сайт має дистрибутиви та посилання на численні модулі, програми, утиліти та додаткову документацію.

Інтерпретатор мови Python може бути розширений функціями і типами даних, розробленими на C або C ++ (або іншою мовою, яку можна викликати за C). Python також зручна як мова розширення для додатків, що вимагають подальшого налагодження.

Розробники мови Python є прихильниками певної філософії програмування, яку називають «The Zen of Python» («Дзен Пайтона») . Її текст можна отримати в інтерпретаторі Python за допомогою команди import this (один раз за сесію).

Автором цієї філософії вважається Тім Пейтерс.

Текст філософії:

- Гарне краще, ніж потворне.
- Явна краще, ніж неявне.
- Просте краще, ніж складне.
- Складне краще, ніж заплутане.
- Плоске краще, ніж вкладене.
- Розріджене краще, ніж щільне.
- Легкість читання має значення.
- Особливі випадки не настільки особливі, щоб порушувати правила.
- При цьому практичність важливіше бездоганності.
- Помилки ніколи не повинні замовчуватися.
- Якщо не замовчуються явно.
- Зустрівши двозначність, відкинь спокуса вгадати.
- Повинен існувати один - і, бажано, тільки один - очевидний спосіб
- 36

- зробити це.
- Хоча спочатку він може бути і не очевидним, якщо ви не голландець [11].
- Зараз краще, ніж ніколи.
- Хоча ніколи, як правило, краще, ніж просто зараз.
- Якщо реалізацію важко пояснити - ідея погана.
- Якщо реалізацію легко пояснити - ідея, можливо, хороша.
- Простір імен - чудова річ! Будемо робити їх побільше!
- Python портовано і працює майже на всіх відомих платформах - від КПК до
- мейнфреймів. Існують порти під Microsoft Windows, всі варіанти UNIX
- (включаючи FreeBSD і GNU / Linux), Plan 9, Mac OS і Mac OS X, iPhone OS 2.0 і
- вище, Palm OS, OS / 2, Amiga, AS / 400 і навіть OS / 390, Symbian і Android [14].

У міру старіння платформи її підтримка в основний гілці мови припиняється. Наприклад, із серії 2.6 припинена підтримка Windows 95, Windows 98 і Windows ME. Однак на цих платформах можна використовувати попередні версії Python - тепер співтовариство активно підтримує версії Python починаючи від 2.3 (для них виходять виправлення). При цьому, на відміну від багатьох портованих систем, для всіх основних платформ Python має підтримку характерних для даної платформи технологій (наприклад, Microsoft COM / DCOM). Більше того, існує спеціальна версія Python для віртуальної машини Java - Jython, що дозволяє інтерпретатору виконуватися на будь-якій системі, яка підтримує Java, при цьому класи Java можуть безпосередньо використовуватися з Python і навіть бути написаними на ньому. Також кілька проектів забезпечують інтеграцію з платформою Microsoft.NET, основні з яких - IronPython і Python.Net. Python, як і багато інших різних мов, не застосовують, наприклад, JIT-компілятори, мають загальний недолік - порівняно невисоку швидкість виконання програм. Однак, у випадку з Python цей недолік компенсується зменшенням часу розробки програми. В середньому

програма, написана на Python, в 2-4 рази компактніше, ніж її аналог на C ++ або Java. Збереження байт-коду (файли .pyc і .pyo) дозволяє інтерпретатору не витрачати зайвий час на перекомпіляцію коду модулів при кожному запуску, на відміну, наприклад, від мови Perl. Крім того, існує спеціальна JIT-бібліотека psyco (проте призводить до збільшення споживання оперативної пам'яті). Ефективність psyco в значній мірі залежить від архітектури програми.

Існують проекти реалізацій мови Python, що вводять високопродуктивні віртуальні машини (ВМ) як компілятора заднього плану. Прикладами таких реалізацій може служити PyPy, заснований на LLVM; більш ранньої ініціативою є проект Parrot. Очікується, що використання ВМ типу LLVM призведе до тих самих результатів, що і використання аналогічних підходів для реалізацій мови Java, де низька обчислювальна продуктивність в основному подолана.

Безліч програм / бібліотек для інтеграції з іншими мовами програмування надають можливість використовувати іншу мову для написання критичних ділянок.

У найпопулярнішій реалізації мови Python інтерпретатор досить великий і більш вимогливий до ресурсів, ніж в аналогічних популярних реалізаціях Tcl, Forth, LISP або Lua, що обмежує його застосування у вбудованих системах. Тим не менш, Python знайшов застосування в КПК і деяких моделях мобільних телефонів

1.8 Конволюційні нейронні мережі

Конволюційні нейронні мережі це спеціальна архітектура штучних нейронних мереж, запропонована Яном Лекуном в 1988 році і націлена на ефективне розпізнавання зображень, входить до складу технологій глибокого навчання. Використовує деякі особливості зорової кори, в якій були відкриті так звані прості клітини, що реагують на прямі лінії під різними кутами, і складні клітини, реакція яких пов'язана з активацією певного набору простих клітин. Таким чином, ідея свёрточних нейронних мереж полягає в чергуванні свёрточних шарів (англ. Convolution layers) і субдіскретізірующіх шарів (англ. Subsampling layers або англ. Pooling layers, шарів підвибірки). Структура мережі - односпрямована (без зворотних зв'язків), принципово багатошарова. Для навчання використовуються стандартні методи, найчастіше метод зворотного поширення помилки. Функція активації нейронів (передавальна функція) - будь-яка, за вибором дослідника. Назва архітектура мережі отримала через наявність операції згортки, суть якої в тому, що кожен фрагмент зображення множиться на матрицю (ядро) згортки поелементно, а результат підсумовується і записується в аналогічну позицію вихідного зображення.

Робота конволюційної нейронної мережі зазвичай інтерпретується як перехід від конкретних особливостей зображення до більш абстрактних деталей, і далі до ще більш абстрактних деталей до виділення понять високого рівня. При цьому мережа самонастроюється і виробляє сама необхідну ієрархію абстрактних ознак (послідовності карт ознак), фільтруючи незначні деталі і виділяючи істотне.

Подібна інтерпретація носить скоріше метафоричний або ілюстративний характер. Фактично «ознаки», що виробляються складною мережею, малозрозумілі і важкі для інтерпретації настільки, що в практичних системах не дуже рекомендується намагатися зрозуміти зміст цих ознак або намагатися їх «підправити», замість цього рекомендується вдосконалити саму структуру і архітектуру мережі, щоб отримати кращі результати. Так, ігнорування системою

якихось істотних явищ може говорити про те, що або не вистачає даних для навчання, або структура мережі має недоліки і система не може виробити ефективних ознак для даних явищ.

У звичайному перцептроном, який представляє собою повнозв'язну нейронну мережу, кожен нейрон пов'язаний з усіма нейронами попереднього шару, причому кожна зв'язок має свій персональний ваговий коефіцієнт. У свёрточной нейронної мережі в операції згортки використовується лише обмежена матриця ваг невеликого розміру, яку «рухають» по всьому оброблюваного шару (на самому початку - безпосередньо по вхідному зображенню), формуючи після кожного зсуву сигнал активації для нейрона наступного шару з аналогічною позицією. Тобто для різних нейронів вихідного шару використовуються одна і та ж матриця ваг, яку також називають ядром згортки. Її інтерпретують як графічне кодування якої-небудь ознаки, наприклад, наявність похилої лінії під певним кутом. Тоді наступний шар, що вийшов в результаті операції згортки такою матрицею ваг, показує наявність даної ознаки в оброблюваному шарі і її координати, формуючи так звану карту ознак (англ. Feature map). Природно, в свёрточной нейронної мережі набір ваг не один, а ціла гама, що кодує елементи зображення (наприклад лінії і дуги під різними кутами). При цьому такі ядра згортки не закладаються дослідником заздалегідь, а формуються самостійно шляхом навчання мережі класичним методом зворотного поширення помилки. Прохід кожним набором ваг формує свій власний примірник карти ознак, роблячи нейронну мережу багатоканальної (багато незалежних карт ознак на одному шарі). Також слід зазначити, що при переборі шару матрицею ваг її пересувають зазвичай не на повний крок (розмір цієї матриці), а на невелику відстань. Так, наприклад, при розмірності матриці ваг 5 × 5 її зрушують на один або два нейрона (пікселя) замість п'яти, щоб не «переступити» шукану ознаку.

Операція Субдискретізація (англ. Subsampling, англ. Pooling, також перекладається як «операція підвибірки» або операція об'єднання), виконує зменшення розмірності сформованих карт признаків. У даній архітектурі мережі вважається, що інформація про факт наявності шуканої ознаки важливіше точного

знання його координат, тому з кількох сусідніх нейронів карти признаків вибирається максимальний і приймається за один нейрон ущільненої карти ознак меншої розмірності. За рахунок цієї операції, крім прискорення подальших обчислень, мережа стає більш інваріантною до масштабу вхідного зображення.

Розглянемо типову структуру конволюційної нейронної мережі більш детально. Мережа складається з великої кількості шарів. Після початкового шару (вхідного зображення) сигнал проходить серію свёрточних шарів, в яких чергується власне згортка і Субдискретізація (пулінг). Чергування шарів дозволяє складати «карти ознак» з карт ознак, на кожному наступному шарі карта зменшується в розмірі, але збільшується кількість каналів. На практиці це означає здатність розпізнавання складних ієрархій ознак. Зазвичай після проходження декількох шарів карта ознак вироджується в вектор або навіть скаляр, але таких карт ознак стають сотні. На виході конволюційних шарів мережі додатково встановлюють кілька шарів повно нейронної мережі (перцептрон), на вхід якого подаються кінцеві карти ознак.

Шар згортки

Шар згортки (англ. Convolutional layer) - це основний блок конволюційної нейронної мережі. Шар згортки включає в себе для кожного каналу свій фільтр, ядро згортки якого обробляє попередній шар за фрагментами (підсумовуючи результати матричного представлення для кожного фрагмента). Вагові коефіцієнти ядра згортки (невеликий матриці) невідомі і встановлюються в процесі навчання.

Особливістю згорткового шару є порівняно невелика кількість параметрів, які встановлюється при навчанні. Так наприклад, якщо вихідне зображення має розмірність 100 × 100 пікселів по трьом каналам (це значить 30000 вхідних нейронів), а згортковий шар використовує фільтри с ядром 3x3 пікселя з виходом на 6 каналів, тоді в процесі навчання визначається тільки 9 ваг ядра, однак по всім

сполученням каналів, тобто 9 × 3 × 6 = 162, в такому випадку даний шар вимагає знаходження тільки 162 параметрів, що істотно менше кількості шуканих параметрів повнозвязної нейронної мережі.

Ректифіковані лінійні одиниці

Ректифіковані лінійні одиниці: цей шар застосовує елементарну функцію активації та вводить нелінійність.

Пулінг або шар Субдіскретізація

Шар пулінг (інакше підвибірки, Субдіскретізація) являє собою нелінійне ущільнення карти ознак, при цьому група пікселів (зазвичай розміру 2 × 2) ущільнюється до одного пікселя, проходячи нелінійне перетворення. Найбільш споживані при цьому функція максимуму. Перетворення зачіпають непересічні прямокутники або квадрати, кожен з яких скорочується в один піксель, при цьому вибирається піксель, що має максимальне значення. Операція ПУЛІНГ дозволяє істотно зменшити просторовий обсяг зображення. Пулінг інтерпретується так. Якщо на попередній операції згортки вже були виявлені деякі ознаки, то для подальшої обробки настільки докладне зображення вже не потрібно, і воно ущільнюється до менш докладного. До того ж фільтрація вже непотрібних деталей допомагає не перенавчатися. Шар пулінгу, як правило, вставляється після шару згортки перед шаром наступної згортки.

Крім пулінгу з функцією максимуму можна використовувати і інші функції, наприклад, середнього значення або L2-нормування. Однак практика показала переваги саме пулінг з функцією максимуму, який включається в типові системи.

З метою більш агресивного зменшення розміру одержуваних уявлень, все частіше знаходять поширення ідеї використання менших фільтрів або повна відмова від шарів пулінгу

Повнозв'язна нейронна мережа

Після кількох проходжень згортки зображення і ущільнення за допомогою пулінг система перебудовується від конкретної сітки пікселів з високою роздільною здатністю до більш абстрактних карт ознак, як правило на кожному

наступному шарі збільшується число каналів і зменшується розмірність зображення в кожному каналі. Зрештою залишається великий набір каналів, що зберігають невелику кількість даних (навіть один параметр), які інтерпретуються як самі абстрактні поняття, виявлені з вихідного зображення.

Ці дані об'єднуються і передаються на звичайну повнозв'язну нейронну мережу, яка теж може складатися з декількох шарів. При цьому повнозв'язні шари вже втрачають просторову структуру пікселів і мають порівняно невеликий розмірністю (по відношенню до кількості пікселів вхідного зображення)

На наступному зображенні (рис. 1.9) наведено типову архітектуру конволюційної нейронної мережі, а саме VGG-16

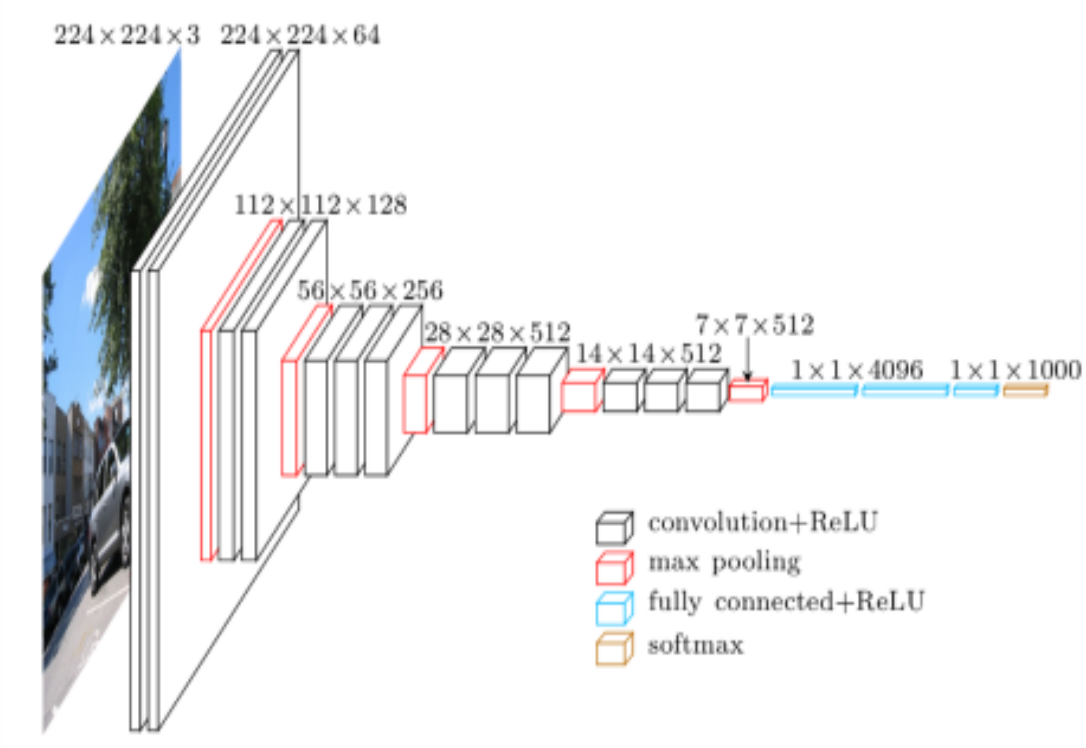


Рисунок 1.9 - Макроструктура конволюційної нейронної мережі VGG-16

Далі наведено структуру мережі VGG-16 у вигляді текстового опису(рис 1.10)

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 224, 224, 3)       0
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 224, 224, 64)      1792
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 224, 224, 64)      36928
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 112, 112, 64)      0
_________________________________________________________________
block2_conv1 (Conv2D)        (None, 112, 112, 128)     73856
_________________________________________________________________
block2_conv2 (Conv2D)        (None, 112, 112, 128)     147584
_________________________________________________________________
block2_pool (MaxPooling2D)   (None, 56, 56, 128)       0
_________________________________________________________________
block3_conv1 (Conv2D)        (None, 56, 56, 256)       295168
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 28, 28, 256)       0
_________________________________________________________________
block4_conv1 (Conv2D)        (None, 28, 28, 512)       1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 14, 14, 512)       0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv2 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv3 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_pool (MaxPooling2D)   (None, 7, 7, 512)         0
_________________________________________________________________
flatten (Flatten)            (None, 25088)             0
_________________________________________________________________
fc1 (Dense)                  (None, 4096)              102764544
_________________________________________________________________
fc2 (Dense)                  (None, 4096)              16781312
_________________________________________________________________
predictions (Dense)          (None, 1000)              4097000
_________________________________________________________________
dense_1 (Dense)              (None, 16)                16016
=================================================================
Total params: 138,373,560
Trainable params: 138,373,560
Non-trainable params: 0
```

Рисунок 1.10 - Стреутура мережі VGG 16

Висновки з розділу

В данному розділі було розлянуто весь використаний інструментарій та всі використовані технології. Також було розлянуто данні їх структуру та всі данні, що використовувались для побудови моделей. Надано короткий опис, для бажаючих ознайомитись з данною роботою

2. АНАЛІЗ ОТРИМАНИХ РЕЗУЛЬТАТІВ.

2.1 Опис використаних алгоритмів

В данній роботі було використано попередньо навченої моделі конволюційних нейронних мереж(опис моделей міститься в розділі 1.7), а саме VGG-16 моделі. Був використаний метод передачі навчання

Передача навчання, як випливає з назви, означає передачу знань, отриманих під час навчання одного CNN, до іншої, але пов'язаної з нею проблеми.

Є два основних сценарії переносу навчання:

1. Виділення признаків

У цьому випадку останній повнозвязний шар видаляється, а решту CNN використовується як екстрактор для нового набору даних.

2. Точна настройка

Тут новий набір даних використовується для точного налаштування ваг попередньо підготовленої CNN. Можна точно налаштувати всі шари або навіть певні шари CNN.

В нашому експерименті було взято мережу vgg-16 для виділення признаків. Далі отримані признаки були відправлено на вхід для навчання та тестування класифікаторів, а саме, на мережу Nefclass(опис структури можна знайти в розділі 1.3). В якості алгоритмів навчання брались 3 алгоритма, а саме, алгоритм «стрибку басейну»(опис алгоритму в розділі 1.6), диференційна еволюція(опис алгоритму в розділі 1.5), метод градієнтного спуску(опис алгоритму в розділі 1.4).

2.2 Опис експериментів

В данній роботі була проведена серія експериментів та порівняна з даними отриманими в роботах попередників. Були використані попередньо навчену модель для виділення признаків. В наступних таблицях (2.1-2.2) будуть представлені результати класифікації з різними параметрами. Данні були розбиті на 2 множини: навчальну та тестову. А саме на 80 да 20 відсотків.

Таблиця 2.1 - Результати мережі NEFCLASS

Початкове розбиття,кількість правил	40X	100X	200X	400X
2,2	73%	74%	74.2%	73.5%
4,2	75.3%	74.8%	75.7%	75.4%
6,2	78.2%	79%	78.4%	78%
8,2	76%	75.4%	76.5%	75.8%
2,4	75%	74%	73.8%	73%
4,4	78.3%	76.3%	75.7%	75.4%
6,4	82%	83%	82.4%	83.2%
8,4	82.2%	81.5%	81.5%	83.8%
2,6	75.4%	73.8%	74.4%	73.2%
4,6	**90%**	**91%**	**90.5%**	**90%**
6,6	89%	89.7%	90.2%	89.5%
8,6	90.3%	90.5%	92%	91.2%
4,8	89.3%	89.8%	89.7%	89.3%
6,8	89.2%	88%	89.4%	88.4%
8,8	88%	87.2%	87.2%	87%

З данної таблиці видно що з експерименту з кількістю початкового розбиття 6, та кількости правил 6, точність не збільшується а складність навчання моделі зростає.

Як бачимо з таблиці для 2-х класів було отримано найкращі значення для кількості розділень 4 та кількості правил 6.

Для порівняння візьмемо данні отримані в попередніх роботах з різними классифікаторами.

Таблиця 2.2 - Порівняння результатів різних класифікаторів

	40X	100X	200X	400X
Linear svm	89%	89%	88%	88%
Polynomial svm	88%	90%	89%	85%
Random forest	89.18%	88%	87.74%	80%
Nefclass	90%	91%	90.5%	90%

В 1-му випадку ми варіювали кількість лінгвістичних змінних та правил, для того що б визначити найкращу, кількість лінгвістичних змінних

Як бачимо нефкласс показує себе трохи краще ніж попередні алгоритми.

Вхідними данними на навчалтьні алгоритми є виділені признаки, а саме 4096 вхідних значень. Признаки виділялись за допомогою попередньо навченої моделі VGG-16, з видаленням останього рішаючого слою.

В данному випадку для навчання мережі нефклас було використано 3 алгоритми навчання, а саме Дифференціальна еволюція, Алгоритм оптимізації «стрибку басейну» та методом найскорішого спуску, вдалось добитись одинакових результатів на «стрибку басейну» та алгоритмі найкоротшого спуску, що може свідчити про вірну оптимальну класификацію. Також з того що алгоритм «стрибку

басейну» пройшов успішно можна зробити висновок що мінімум було знайдено вірно, також це свідчить про те що мінімум досить укомплектований

Також було оскільки після дії екстрактора, вхідних параметрів дуже багато, а саме 4096 – то доцільно було б зменшити їх розмірність, для цього було використано метод головних компонент. В таблиці 2.3 було представвлено результати декомпозиції.

Таблиця 2.3 – Залежність варіації від кількості компонент

Кількість компонент	Варіація	Час навчання(приблизний)
100	0.840587442159	~2 год
200	0.897366730496	~3 год
250	0.912324353994	~4 год
500	0.954868534936	~9 год

З таблиці ми бачимо що результат декомпозиції з кількістю 250 найбільш задовільний в звязку з тим що складність навчання зростає пропорційно зі збільшенням вхідних данних. За недостатньою кількістю часу дальші експерименти було вирішено проводити на данних зі 100 кратним збільшенням(2081 зображення). В наступній таблиці ми бачимо результати перевірки (табл 2.4)

Початкове розбиття,кількість правил	100X
4,4	80.64%
4,6	87.24%
4,8	88.18%

Таблиця 2.4 - Точність експерименту при кількості признаків рівній 250

З цієї таблиці ми бачимо що точність в данних експериментах впала на кілька відсотків, через те що ми декомпозицією відкинули признаки, проте ми значно скоротили час навчання.

Таблиця 2.5 – Точність експерименту при різних кількостях признаків

Початкове розбиття,кількість правил	100	250	4096
4,4	75.23%	80.64%	76.3%
4,6	83.34%	87.24%	91%
4,8	84.21%	88.18%	89.8%

З даної таблиці (табл 2.5) ми можемо бачити, що точність паде зі зменшенням кількості признаків, проте не так суттєво в середньому на 3-5% відсотка якщо порівнювати 100 та 250 признаків. Для порівняння візьмемо 4096, ми бачимо що зміни кільсті признаків в 20 разів точність паде незначно в середньому на 2-3% відсотка. Що може свідчити про незначний вплив признаків на класифікацію.

Також хочу звернути увагу на те, що в звязку з недостатньою кількістю часу була проведена недостатня кількість експериментів, можливо попрацювавши з налаштуванням, можна було б отримати й кращий результат.

З цього ми можемо зробити виснвок що для класифікації медичних зображень на тип пухлини підходить модель після декомпозиції.

Висновки до розділу

З результату роботи ми можемо побачити що мережа nefclass показала себе в задачі класифікації краще за використані в попередніх роботах в середньому на 2 відсотки що свідчить про підходящість данної моделі для данних задач.

Також використання методу головних компонент дало позитивний результат до швидкодії, проте за нестачі часу не вдалось прогнати достатню кількість експериментів.

В цілому вдалося добитись результатів 91%, що краще за попередньо отримані результати на 1-2%.

3. РОЗРОБЛЕННЯ СТАРТАП-ПРОЕКТУ

3.1 Інформаційна карта проекту

Таблиця 3.1 - Інформаційна карта проекту

1. Назва проекту	Рабочее название
2. Автори проекту	Варга
3. Коротка анотація	Алгоритм
4. Термін реалізації проекту	18 місяців
5. Необхідні ресурси	TyanGT20B7002 — 1 шт ~ 700$, процессор Intel «XeonE5620» — 2 шт ~ 760$, жёсткие диски WD «VelociRaptorWD1500HLFS» — 4 шт ~ 480$ 2 GbECCKingston — 8 шт ~ 540$ 200000$ витрати на персонал 20000$ витрати на приміщення

Продовження таблиці 3.1

6. Головні цілі та завдання проекту	Створення десктопного додатку для особистого користування та окремого продукту для промислового встановлення. Привернення уваги технологічних корпорацій до нашої команди для інвестицій у розробки більш важливих проектів, потребуючих великого фінансування.
7. Очікувані результати	Збільшення зацікавленності використання нейронних мереж компаніями, які не є міжнародними корпораціями. Привернення уваги технологічних корпорацій до нашої команди для інвестицій у розробки більш важливих проектів, потребуючих великого фінансування Збір інформації

Команда стартапу

Таблиця 3.2 - Команда проекту

Керівник проекту	Пошук інвесторів, керування компанією
Технічній директор	Керування технічною стороною проекту, розробка архітектури, підбір необхідних технологій. Досвід у програмуванні та менеджменті.
Програміст (2)	Розробка додатку. Досвід у машинному навчанні. Досвід створення серверних додатків.
Маркетолог	Розробка стратегій зацікавлення потенційних користувачів.Досвід популярізації продукту.
Веб-розробник	Створення сайту-платформи для взаємодії з серверним додатком. Навички у дизайні та веб-розробці.

3.2 Маркетингова стратегія та маркетинговий план стартапу

3.2.1 Опис ідеї продукту

В межах підпункту послідовно проаналізовані та подані у вигляді таблиць:

- зміст ідеї (що пропонується);
- можливі напрямки застосування;
- основні вигоди, що може отримати користувач товару (за кожним напрямком застосування);
- чим відрізняється від існуючих аналогів та замінників;

Перші три пункти подані у вигляді таблиці (табл. 5.1) і дають цілісне уявлення про зміст ідеї та можливі базові потенційні ринки, в межах яких потрібно шукати групи потенційних клієнтів.

Таблиця 3.3 - Опис ідеї стартап-проекту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Додаток, в основі якого йдуть алгоритми глибинного навчання, що допомагають ракові клітини. На основі моделі користувач зможе отримати доступ до аккаунту, аналіз емоцій та активності обличчя та настрій людини.	1. Діагностика	Для швидкої діагностики
	2. Програма - аналізатор	Програма зможе класифікувати тканини людини.

Аналіз потенційних техніко-економічних переваг ідеї (чим відрізняється від існуючих аналогів та замінників) порівняно із пропозиціями конкурентів передбачає:

- визначення переліку техніко-економічних властивостей та характеристик ідеї
- визначення попереднього кола конкурентів (проектів-конкурентів) або товарів-замінників чи товарів-аналогів, що вже існують на ринку, та проводиться збір інформації щодо значень техніко-економічних показників для ідеї власного проекту та проектів-конкурентів відповідно до визначеного вище переліку;
- проводиться порівняльний аналіз показників: для власної ідеї визначаються показники, що мають а) гірші значення (W, слабкі); б) аналогічні (N, нейтральні) значення; в) кращі значення (S, сильні) (табл. 5.2).
-

Таблиця 3.4 - Визначення сильних, слабких та нейтральних характеристик ідеї проекту

	Техніко-економічні характеристики ідеї	(потенційні) товари/концепції конкурентів			W (слабка сторона)	N (нейтральна сторона)	S (сильна сторона)
		Наш продукт	Конкурент 1	Конкурент 2			
	Швидкість роботи						+

Продовження таблиці 3.4

	Інтерфейс					+	
	Класифікація тканин						+

Визначений перелік слабких, сильних та нейтральних характеристик та властивостей ідеї потенційного товару є підґрунтям для формування його конкурентоспроможності.

3.2.1.1 Технологічний аудит ідеї продукту

В межах даного підрозділу необхідно провести аудит технології, за допомогою якої можна реалізувати ідею проекту (технології створення товару). Визначення технологічної здійсненності ідеї проекту передбачає аналіз таких складових (табл. 5.3):

за якою технологією буде виготовлено товар згідно ідеї проекту?

чи існують такі технології, чи їх потрібно розробити/доробити?

чи доступні такі технології авторам проекту?

Таблиця 3.5 - Технологічна здійсненність ідеї проекту

	Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій
1	ML	Caffe	Є у наявності	Відкрите програмне забезпечення
2	ML	Torch7	Є у наявності	Відкрите програмне забезпечення
3	ML	Tensorflow	Є у наявності	Відкрите програмне забезпечення
4	Database	MongoDB	Є у наявності	Відкрите програмне забезпечення
5	Database	PostgreSQL	Є у наявності	Відкрите програмне забезпечення
Обрана технологія Caffe + MongoDB				

За результатами аналізу таблиці робиться висновок щодо можливості технологічної реалізації проекту: так чи ні, а також технологічного шляху, яким це доцільно зробити (з поміж названих технологій обираються такі, що доступні авторам проекту та є наявними на ринку).

3.4 Аналіз ринкових можливостей запуску стартап-проекту

Визначення ринкових можливостей, які можна використати під час ринкового впровадження проекту, та ринкових загроз, які можуть перешкодити реалізації проекту, дозволяє спланувати напрями розвитку проекту із урахуванням стану ринкового середовища, потреб потенційних клієнтів та пропозицій проектів-конкурентів.

Таблиця 3.6 - Попередня характеристика потенційного ринку стартап-проекту

	Показники стану ринку (найменування)	Характеристика
1	Кількість головних гравців, од	7
2	Загальний обсяг продаж, грн/ум.од	5$/ум.од
3	Динаміка ринку (якісна оцінка)	Стабільна
4	Наявність обмежень для входу (вказати характер обмежень)	Немає
5	Специфічні вимоги до стандартизації та сертифікації	Немає
6	Середня норма рентабельності в галузі (або по ринку), %	48%

Спочатку проводиться аналіз попиту: наявність попиту, обсяг, динаміка розвитку ринку (табл. 5.4).

Рентабельність — поняття, що характеризує економічну ефективність виробництва, за якої за рахунок грошової виручки від реалізації продукції (робіт, послуг) повністю відшкодовує витрати на її виробництво й одержується прибуток як головне джерело розширеного відтворення. Суть одного із найважливіших методів оцінки економічної ефективності інвестицій полягає у розрахунку їх середньої рентабельності за формулою

$$R = P/\ 1 * n * \ 100, \tag{8}$$

де Р - прибуток за час експлуатації проекту; / - повна сума інвестиційних витрат; п - час експлуатації проекту.

Інвестувати грошові засоби доцільно тоді, коли від цього можна отримати більший прибуток, ніж від їх зберігання. Порівнюючи середньорічну рентабельність інвестицій зі ставкою банківського відсотка, можна дійти висновку, що вигідніше.

Середня норма рентабельності в галузі (або по ринку) порівнюється із банківським відсотком на вкладення. За умови, що останній є вищим, можливо, має сенс вкласти кошти в інший проект.

За результатами аналізу таблиці робиться висновок щодо того, чи є ринок привабливим для входження за попереднім оцінюванням.

Надалі визначаються потенційні групи клієнтів, їх характеристики, та формується орієнтовний перелік вимог до товару для кожної групи (табл. 5.5).

Таблиця 3.7 - Характеристика потенційних клієнтів стартап-проекту

Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
Потреба у аналізі емоцій співбесідника, ташвидкої аутентифікації	Компанії, які потребують ідентифікації людей, аналізу аудиторій у певних місцях, окремі користувачі	Різні потреби у аналізі. Наявність або відсутніх необхідних даних для ідентифікації	Кросс-платформенне рішення, доступне для усіх, для початку Робота онлайн Веб-сервіс

Після визначення потенційних груп клієнтів проводиться аналіз ринкового середовища: складаються таблиці факторів, що сприяють ринковому впровадженню проекту, та факторів, що йому перешкоджають (табл. №№ 5.6- 5.7). Фактори в таблиці подаються в порядку зменшення значущості.

Таблиця 3.8 - Фактори загроз

Фактор	Зміст загрози	Можлива реакція компанії
Конкуренція	Збільшення зацікавленності великих гравців в цій галузі	Розробка унікального продукту, який буде більш якісним та мати цікаві нововведення
Зміна потреб користувачів	Зміна пріорітетів користувачів, у зв'язку з якими, програмне забезпечення перестає задовольняти клієнта	Розширення функціональних можливостей додатку.
Низькі продажі	Додаток не затребуваний	Збільшити трати на рекламу
Низький ріст продажів	Погано поширюється	Розробити вірусну рекламу

Таблиця 3.9 - Фактори можливостей

Фактор	Зміст можливості	Можлива реакція компанії
Увага компаній	Привернення увагивеликих компаній до нашої команди	Розширення команди та початокроботи над дорогими проектами

Продовження таблиці 3.9

Висока популярність додатку	Увага аудиторії до продукту дозволяє пропонувати їм нові продукти	Пропозиції укласти контракту із зацікавленними у розширенні аудиторії за рахунок реклами в нашому додатку

Надалі проводиться аналіз пропозиції: визначаються загальні риси конкуренції на ринку (табл. 3.10)

Таблиця 3.10 - Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
1. Вказати тип конкуренції - монополія/олігополія/ монополістична/чиста	Чиста	Ніякого. Клієнти отримують додаток безкоштовно - компанія отримує прибуток з реклами. Клієнти які купують додаток приносять прямий прибуток.

Продовження таблиці 3.10

2. За рівнем конкурентної боротьби - локальний/національний/…	Світовий	Сама природа мобільного додатку дозволяє пропонувати продукт всьому світові без затрат
3. За галузевою ознакою - міжгалузева/ внутрішньогалузева	Внутрішньогалузева	Велика кількість інших розроблених додатків в яких наш може просто загубитися, тому необхідно розробити якір для уваги — слоган, назву, логотип додатку
4. Конкуренція за видами товарів: - товарно-родова - товарно-видова - між бажаннями	Між бажаннями	Вкладення в різні види реклами, щоб переконати клієнта, що йому потрібен наш продукт

Продовження таблиці 3.10

5. За характером конкурентних переваг - цінова / нецінова	Нецінова	Конкурентні переваги — функції та можливості додатку
6. За інтенсивністю - марочна/не марочна	Марочна	Велика кількість інших розроблених додатків в яких наш може просто загубитися, тому необхідно розробити якір для уваги — слоган, назву, логотип додатку

Таблиця 3.11 - Аналіз конкуренції в галузі за М. Портером

Складові аналізу	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари-замінники
	Навести перелік прямих конкурентів	Визначити бар'єри входження в ринок	Визначити фактори сили постачальників	Визначити фактори сили споживачів	Фактори загроз з боку замінників
Висновки	Визначити інтенсивність конкурентної боротьби з боку прямих конкурентів	- чи є можливості входу в ринок? - чи є потенційні конкуренти? Строки виходу їх на ринок?	Чи постачальники диктують умови роботи на ринку? Які?	Чи клієнти диктують умови роботи на ринку? Які?	Обмеження для роботи на ринку через товари замінники

За результатами аналізу таблиці робиться висновок щодо принципової можливості роботи на ринку з огляду на конкурентну ситуацію. Також робиться висновок щодо характеристик (сильних сторін), які повинен мати проект, щоб бути конкурентоспроможним на ринку. Другий висновок враховується при формулюванні переліку факторів конкурентоспроможності у п. 3.6.

На основі аналізу конкуренції, проведеного в п. 3.5 (табл. 5.9), а також із урахуванням характеристик ідеї проекту (табл. 5.2), вимог споживачів до товару (табл. 5.5) та факторів маркетингового середовища (табл. №№ 5.6-5.7) визначається та обґрунтовується перелік факторів конкурентоспроможності. Аналіз оформлюється за табл. 5.10

Таблиця 3.12 - Обґрунтування факторів конкурентоспроможності

Фактор конкурентоспроможності	Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)
Якість продукту	Один із факторів для вибору продукту клієнтом.
Кількість видів соціальної взаємодії	Більша зацікавленість клієнта продуктом.
Ціна	Один із факторів для вибору продукту клієнтом.
Простота експлуатації	Один із факторів для зберігання зацікавленності клієнта.

За визначеними факторами конкурентоспроможності (табл. 5.10) проводиться аналіз сильних та слабких сторін стартап-проекту (табл. 5.11).

Таблиця 3.13 - Порівняльний аналіз сильних та слабких сторін Emozzion

	Фактор конкурентоспроможності	Бали 1-20	Рейтинг товарів-конкурентів у порівнянні						
			3	2	1				
	Якість продукту	15							
	Різноманітя функціоналу	10							
	Ціна	10							
.	Простота експлуатації	15							

Фінальним етапом ринкового аналізу можливостей впровадження проекту є складання SWOT-аналізу (матриці аналізу сильних (Strength) та слабких (Weak) сторін, загроз (Troubles) та можливостей (Opportunities) (табл.5. 12) на основі виділених ринкових загроз та можливостей, та сильних і слабких сторін (табл. 5.11).

Перелік ринкових загроз та ринкових можливостей складається на основі аналізу факторів загроз та факторів можливостей маркетингового середовища. Ринкові загрози та ринкові можливості є наслідками (прогнозованими 75 результатами) впливу факторів, і, на відміну від них, ще не є реалізованими на ринку та мають певну ймовірність здійснення. Наприклад: зниження доходів потенційних споживачів – фактор загрози, на основі якого можна зробити прогноз щодо посилення значущості цінового фактору при виборі товару та відповідно, – цінової конкуренції (а це вже – ринкова загроза).

Таблиця 3.14 - SWOT- аналіз стартап-проекту

Сильні сторони: ціна, простота використання	Слабкі сторони: досі невідома компанія
Можливості: наявність безкоштовного функціоналу	Загрози: видавлення з ринку конкурентами, не розуміння користувачем переваг

На основі SWOT-аналізу розробляються альтернативи ринкової поведінки (перелік заходів) для виведення стартап-проекту на ринок та орієнтовний оптимальний час їх ринкової реалізації з огляду на потенційні проекти конкурентів, що можуть бути виведені на ринок (див. табл. 9, аналіз потенційних конкурентів).

Визначені альтернативи аналізуються з точки зору строків та ймовірності отримання ресурсів (табл. 5.13). З означених альтернатив обирається та, для якої: а) отримання ресурсів є більш простим та ймовірним; б) строки реалізації – більш стислими.

Таблиця 3.15 - Альтернативи ринкового впровадження стартап-проекту

Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
PR, просування бренду	50%	5
Перехід на безкоштовне розповсюдження	65%	3
Партнерство для об’єднання продукції	75%	3

3.3 Розроблення ринкової стратегії продукту

Розроблення ринкової стратегії першим кроком передбачає визначення стратегії охоплення ринку: опис цільових груп потенційних споживачів (табл. 3.16)

Таблиця 3.16 - Вибір цільових груп потенційних споживачів

№	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1	Поодинокі користувачі	Висока	Середній	Невисока	Низька
2	Малі компанії, зацікавлені в аналізі аудиторії людей.	Висока	Високий	Невисока	Середня
3	Крупні компанії,	Невисока	Невисокий	Висока	Середня

За результатами аналізу потенційних груп споживачів (сегментів) автори ідеї обирають цільові групи, для яких вони пропонуватимуть свій товар, та визначають стратегію охоплення ринку:

- якщо компанія зосереджується на одному сегменті – вона обирає стратегію концентрованого маркетингу;

- якщо працює із кількома сегментами, розробляючи для них окремо програми ринкового впливу – вона використовує стратегію диференційованого маркетингу;

- якщо компанія працює із всім ринком, пропонуючи стандартизовану програму (включно із характеристиками товару/послуги) – вона використовує масовий маркетинг.

Для роботи в обраних сегментах ринку необхідно сформувати базову стратегію розвитку (табл. 5.15).

Стратегія диференціації передбачає надання товару важливих з точки зору споживача відмітних властивостей, які роблять товар відмінним від товарів конкурентів. Така відмінність може базуватися на об'єктивних або суб'єктивних, відчутних і невідчутних властивостях товару(у ширшому розумінні – комплексі маркетингу), бути реальною або уявною. Інструментом реалізації стратегії диференціації є ринкове позиціонування.

Реалізація цієї стратегії вимагає, як правило, більш високих витрат. Проте успішна диференціація дозволяє компанії домогтись більшої рентабельності за рахунок того, що ринок готовий прийняти більш високу ціну (цінову премію бренду).

При веденні конкурентної боротьби з використанням цієї стратегії на ринку в першу чергу терплять фіаско фірми, що не здатні визначати потреби цільових ринків, оперативно реагувати на зміни в ринковому попиті, проводити ефективну політику маркетингових комунікацій, не мають необхідних навичок в області брендингу. Найважливішими здібностями, які повинна мати компанія, що приймає цю стратегію, є з генерування маркетингових ноу-хау, здійснення продуктових новацій.

3.4 Розроблення маркетингової програми стартап-проекту

Таблиця 3.17 - Визначення базової стратегії розвитку

Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
Надання товару важливих з точки зору споживача відмітних властивостей	Стратегія диференціації	Більша кількість способів соціальної взаємодії, можливість переходу до безкоштовного типу розповсюдження	Стратегія диференціації

Наступним кроком є вибір стратегії конкурентної поведінки.

Стратегія наслідування лідеру

Компанії, що приймають слідування за лідером – це підприємства з невеликою часткою ринку, які вибирають адаптивну лінію поведінки на ринку, усвідомлюють своє місце на нім і йдуть у фарватері фірм-лідерів. Головна перевага такої стратегії – економія фінансових ресурсів, пов’язаних з необхідністю розширення товарного(галузевого) ринку, постійними інноваціями, витратами на утримання домінуючого положення

Стратегія наслідування лідеру найчастіше має місце у випадку олігополії, коли кожен конкурент прагне уникнути боротьби, особливо цінової, а також у випадку, коли слабо виражений ефект масштабу, що не дозволяє отримати переваги від об’ємів продажів або ж він не грає істотної ролі. Стратегію наслідування лідеру приймають також фірми, які не змогли реалізувати стратегію виклику лідерові.

Компанії, що приймають таку стратегію, зазвичай випускають товари-імітатори, займаючи ринкову частку, яку з різних причин не можуть охопити фірми лідери. Вибір такої стратегії може також бути обумовлений також перевагою локалізації (краще знання ринку, налагоджені зв'язки з клієнтами тощо).

Для ефективної реалізації цієї стратегії компанії повинні задовольняти наступним основним умовам:

- систематичний аналіз сегментації ринку з метою виділення нових ринкових сегментів або таких, що незадовільно обслуговуються;

- ефективне використання НДДКР з метою вдосконалення технологічних процесів і незначних продуктових новацій;

- концентрація на прибутковості, а не на простому зростанні об'ємів продажів;

- постійний аналіз витрат на всіх стадіях виробництва і логістики;

- залишатися досить малим, щоб не бути досить цікавим для фірм-лідерів;

- сильний керівник, здатний не лише формулювати стратегію, але і тримати усю діяльність компанії під власним контролем.

Якщо врахувати, що лідерами ринку можуть бути лише декілька компаній, то ця стратегія наймасовішою.

Таблиця 3.18 - Визначення базової стратегії конкурентної поведінки

Чи є проект «першопрохідцем» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки
Ні	Забирати існуючих	Ні, він буде їх повязувати та розширювати, створюючи новий функціонал	Стратегія наслідування лідеру

На основі вимог споживачів з обраних сегментів до постачальника (стартап-компанії) та до продукту (див. табл. 3.7), а також в залежності від обраної базової стратегії розвитку (табл. 3.17) та стратегії конкурентної поведінки (табл. 3.18) розробляється стратегія позиціонування (табл. 3.19). що полягає у формуванні ринкової позиції (комплексу асоціацій), за яким споживачі мають ідентифікувати торгівельну марку/проект.

Таблиця 3.19 - Визначення стратегії позиціонування

Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)
Якість	Позиціювання за показниками якості	Тестування розробленого продукту та виправлення всіх багів	Стабільність роботи, якість роботи
Більша кількість функціональних можливостей	На основі специфічних відчутних характеристик	Розробка більшої кількості оригінальних можливостей	Велика можливостей

Результатом виконання підрозділу має стати узгоджена система рішень щодо ринкової поведінки стартап-компанії, яка визначатиме напрями роботи стартап-компанії на ринку.

Висновки до розділу

В даному розділі було проведено аналіз програмного продукту у якості стартап проекту. Можна зазначити що у проекті є можливість комерціалізації, адже ринок технологій який базується на розпізнання облич динамічно розвивається, створюються нові сервіси які, одним з якиї і являється наш продукт.

На локальному ринку наявна незначна конкуренція, тому вихід на нього не буде важким. Проект є доволі конкурентноспроможним завдяки, в першу чергу, своїй невисокій вартості, проте для реалізації проекту необхідно буде залучати певні інвестиції для розширення штату робітників, оренди серверів, оренди офісу, купівлі потужного обладнання, яке необхідне для коректної роботи сервісу .

Для впровадження ринкової реалізації проекту слід обрати альтернативу, яка передбачає розробку програмного продукту, а потім якісну рекламу та PR, сконцентровану навколо позитивних характеристиках даного програмного продукту, таких як низька ціна, кросплатформеність тощо.

ВИСНОВКИ

Основним завданням даної роботи було розробити та дослідити діючу модель для класифікації медичних зображень ракових тканей. Було досліджено вплив використання попередньо навченої моделі, та зроблено висновок що вона підійшла для класифікації двух типів пухлин а саме доброякісних та злоякісних.

В ході її виконання були отримані наступні науково-практичні результати:

1. Використання попередньо навченої моделі підходить для визначення типу клітини, проте не підходить для визначення підтипу
2. Експерименти показали, що із проаналізованих варіантів та порівнянь зі зробленими попередньо конфігураціями мереж, найкраще результати показала мережа нефклас

Також можливі наступні варіанти розвитку данного продукту, а саме:

1. Використати інші алгоритми навчання
2. Використати навчання на відеокарті для оптимізації швидкості
3. Використати інші види класифікаторів
4. Спробувати використати інші види декомпозиції
5. Спробувати підкоригувати систему ваг на мережі VGG-16, тобто перенавчити її, що допоможе класифікувати данні більш точно, а саме визначати підтип пухлини

На основі данних вдалося навчити мережі на данних молочної залози та визначити базовий тип пухлини з точністю не менш ніж 90 відсотків. Проте варто відзначити, що час навчання мережі дуже значний, й конкретна реалізація алгоритму не оптимізована під многопоточну архітектуру, також було б добре відзначити що сучасні алгоритми навчаються на відеокартах, якщо будувати модель на відеокарті, то можна було б провести значно більшу кількість експериментів.

ПЕРЕЛІК ПОСИЛАНЬ

1. Krizhevsky A., Imagenet classification with deep convolutional neural networks / Krizhevsky A., Sutskever I., Hinton G. E. // Advances in neural information processing systems. – 2012. – pp. 1097-1105.
2. P. A. Viola. Rapid object detection using a boosted cascade of simple features. / P. A. Viola, M. J. Jones. // In Proc. IEEE Conference on Computer Vision and Pattern Recognition. – 2001.
3. Dalal N. Histograms of oriented gradients for human detection. Computer Vision and Pattern Recognition / Dalal N., Triggs B. // IEEE Computer Society Conference on. – 2005. – vol. 1. – pp. 886-893.
4. Olszewska J. Automated Face Recognition: Challenges and Solutions / Olszewska J. // Intech – Open science Open minds , [Електроннийресурс]. – Режимдоступу: https://cdn.intechopen.com/pdfs-wm/52911.pdf - /. –Дата доступу : 31.05.2017
5. Sirovich L. Low-dimensional procedure for the characterization of human / Sirovich L., Kirby M. // Journal of the Optical Society of America. Optics, Image Science and Vision. – 1987 – 4(3) – pp. 519–524.
6. Face recognition: A literature survey. /[Zhao W., Chellappa R., Rosenfeld A., Phillips P.J.]. // ACM Computing Surveys. - 2003. - 35(4).- pp. 399–458.
7. Iosifidis, A. Scaling-up class-specific kernel discriminant analysis for large-scale face verification. / Iosifidis, A., Gabbouj, M. // IEEE Transactions on Information Forensics and Security. – 2016. – 11(11) – pp. 2453 2465
8. Almudhahka N. Human face identification via comparative soft biometrics./ Almudhahka N., Nixon M., Hare J. // In: Proceedings of the IEEE International Conference on Identity, Security and Behavior Analysis (ISBA) – Sendai, JP – 29 Feb – 02 Mar 2016 – pp. 1–6.
9. Who's in the Picture? /[Berg T.I., Berg A.C., Edwards J., Forsyth D.A.] // In: Proceedings of the Neural Information Processing Systems Conference (NIPS) –

2004 – pp. 137–144

10. Torres L. Is there any hope for face recognition? / Torres L. // In: Proceedings of the IEEE International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS). – 2004.
11. Nash S. Interactively-pose-corrected face recognition. / Nash S., Rhodes M., Olszewska J. I. // In: Proceedings of the INSTICC International Conference on Bio-Inspired Systems and Signal Processing (BIOSIGNALS) – 2016 – pp.106–112.
12. Macro- and micro-expression spoϱing in long videos using spatio temporal strain. / [Shreve M., Godavarthy S., Goldgof D., Sarkar, S.] // In: Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition Workshops(AFGR) – 2011 – pp. 51–56.
13. The Face Image Format Standards : ISO/IEC 19794-5:2011 [Електронний ресурс] – Режим доступу: https://www.iso.org/standard/50867.html
14. Towards face recognition at a distance. / [Prince S.J.D., Elder J., Hou Y. et al.] // In: Proceedings of the IET Conference on Crime and Security. – 2006. – pp. 570–575.
15. Mudunuri S.P. Low resolution face recognition across variations in pose and illumination. / Mudunuri S.P., Biswas S. // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 2016 – 38(5) – pp. 1034–1040.
16. Zou W.W.W. Very low resolution face recognition problem. / Zou W.W.W., Yuen P.C. // IEEE Transaction on Image Processing. – 2012. – 21(1) – pp. 327–340.
17. Huang H., He H. Super resolution method for face recognition using non-linear mappings on coherent features. / Huang H., He H. // IEEE Transaction on Neural Networks. – 2011 – 22(1) – pp. 121–130.
18. Li H. Guided iterative back-projection scheme for single image super-resolution. / Li H., Lam K.M. // In: Proceedings of the IEEE Global High Tech Congress on Electronics (GHTCE). – 2013 – pp. 175–180.
19. Senior A.W. Privacy protection and face recognition. Handbook of Face

Recognition. 2nd ed / Senior A.W, Pankanti S.; in: Li S.Z., Jain A.K.. editors.. – London:Springer, 2011. – 693 p.

20. The FERET evaluation methodology for face recognition algorithms. / [Phillips P.J., Moon H., Rizvi S.A., Rauss P.J.] // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 2000 – 22(10) – pp. 1090–1104.
21. Jesorsky O. Face detection using the Hausdorff distance. / Jesorsky O., Kirchberg K., Frischholz R. in Bigun J., Smeraldi F., editors. // Audio and Video based Person Authentication. LNCS – Springer. 2001 – pp. 90 95.
22. Caltech 10000 Web Faces. Human Faces Collected from Google Image Search. CalTech University. Pasadena, California, USA [Internet]. 2005. [Електронний ресурс]. – Режим доступу: www.vision.cal- tech. edu/Image_Datasets/Caltect_10K_WebFaces. /. – Дата доступу : 21.05.2017.
23. Caltech 10000 Web Faces. Human Faces Collected from Google Image Search. CalTech University. Pasadena, California, USA [Електронний ресурс]. – Режим доступу: www.vision.cal-tech.edu/Image_Datasets/ Caltect_10K_ WebFaces. – Датадоступу: 02.05.2017
24. Overview of the face recognition grand challenge. / [Phillips P.J., Flynn P.J., Scruggs T. et al] // In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) – 2005 – I – pp. 947–954.
25. Yale Face Database. Yale University. Connecticut, USA [Електроннийресурс]. – Режимдоступу: http://vision.ucsd.edu/yale_face_dataset_original /yalefaces.zip [Accessed: 2016-07-07] . – Датадоступу : 03.05.2017.
26. Comparison of face verification results on the XM2VTS database. / [Matas J., Hamouz M., Jonsson K. et al] // In: Proceedings of the IEEE International Conference on Pattern Recognition (ICPR). – 2000. – pp. 858– 863.
27. SCface - Surveillance cameras face database. / [Grgic M., Delac K., Grgic S., Klimpak B.] // Multimedia Tools Applications. – 2011 – 51 – pp. 863–879.
28. FDDB. A Benchmark for Face Detection in Unconstrained Settings Jain V., Learned - Miller E. Technical Report. UM-CS-2010-009. University of Massachusetts Amherst. USA [Електронний ресурс]. – Режим доступу:

http://vis-www.cs.umass.edu/fddb. Дата доступу: 03.05.2017

29. Labeled Faces in the Wild: A Survey. / [Learned-Miller E., Huang G.B., Roy-Chowdhury A. et al] : In: Kawulok M., Emre Celebi M., Smolka B., editors // Advances in Face Detection and Facial Image Analysis. – Springer 2016 – pp. 189–248.
30. A spontaneous micro-expression database: Inducement, collection and baseline. / [Li X., Pefister T., Huang X. et al] // In: Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition (AFGR). – 2013 – pp. 1–6.
31. Выбор функции активации обучения нейронной сети [Електронний ресурс]. – Режим доступу: https://monographies.ru/ru/book/section?id=2465. – Дата доступу : 03.05.2017.
32. Manisha M. Face Recognition Using Neural Network: A Review// Manisha M.Kasar1 , Debnath Bhattacharyya1 and Tai-ho Kim2. [Електронний ресурс]. – Режим доступу: http://www.sersc.org/journals/IJSIA/vol10_no3_2016/8.pdf - – Дата доступу : 03.06.2017.
33. Хемант Синг Миттал, Гарприт Каур Распознавание с использованием метода главных компонент и нейросети. – 2013.[Електронний ресурс].– Режим доступу: http://www.ijese.org/attachments/File/v1i6/F0266 041613.pdf – Дата доступу :04.06.2017.
34. N Jindal. Enhanced Face Recognition Algorithm using PCA with Artificial Neural Networks. / N Jindal, V Kumar // International Journal of Advanced Research in Computer Science and Software Engineering – 2013 – vol 3 pp. 864-872.
35. S-H Yooa. Optimized face recognition algorithm using radial basis function neural networks and its practical applications. / S-H Yooa, S-K Oha, Witold Pedrycz // International journal on Neural Networks– 2015 – vol 69 – pp. 111-125.
36. A Convolutional Neural Network Cascade for Face Detection. / [H Liy, Z Linz, X Shenz et al.] // IEEE Conference on Computer Vision and Pattern Recognition – Boston, MA – 2015 – pp. 5325 – 5334.

37. M.Nandini. Face Recognition Using Neural Network. / M.Nandini, P.Bhargavi, G.Raja Sekhar // International Journal of Scientific and Research Publications – 2013 – vol 3 – pp. 1-5.

38. H A. Rowley. Neural Network-Based Face Detection. / H A. Rowley. // IEEE Computer Society Conference on Computer Vision and Pattern Recognition – San Francisco, CA – 1996) – pp. 203 – 208.

39. P. Boyle and B. Levin, Eds., World Cancer Report 2008. Lyon: IARC, 2008. [Online]. Available: http://www.iarc.fr/en/publications/ pdfs-online/wcr/2008/wcr_2008.pdf

40. Olson, B., Hashmi, I., Molloy, K., and Shehu1, A., Basin Hopping as a General and Versatile Optimization Framework for the Characterization of Biological Macromolecules, Advances in Artificial Intelligence, Volume 2012 (2012), Article ID 674832

41. Wales, David J. 2003, Energy Landscapes, Cambridge University Press, Cambridge, UK.

42. S. R. Lakhani, E. I.O., S. Schnitt, P. Tan, and M. van de Vijver, WHO classification of tumours of the breast, 4th ed. Lyon: WHO Press, 2012.

43. B. Stenkvist, S. Westman-Naeser, J. Holmquist, B. Nordin, E. Bengtsson, J. Vegelius, O. Eriksson, and C. H. Fox, “Computerized nuclear morphometry as an objective method for characterizing human cancer cell populations,” Cancer Research, vol. 38, no. 12. 1978, pp. 4688–4697,.

44. M. Kowal, P. Filipczuk, A. Obuchowicz, J. Korbicz, and R. Monczak, “Computer-aided diagnosis of breast cancer based on fine needle biopsy microscopic images,” Computers in Biology and Medicine, vol. 43, no. 10, pp. 1563–1572, 2013.

45. P. Filipczuk, T. Fevens, A. Krzy˙zak, and R. Monczak, “Computer-aided breast cancer diagnosis based on the analysis of cytological images of fine needle biopsies,” IEEE Transactions on Medical Imaging, vol. 32, 45 no. 12, pp. 2169–2178, 2013.

46. Y. M. George, H. L. Zayed, M. I. Roushdy, and B. M. Elbagoury,"Remote computer-aided breast cancer detection and diagnosis system based on cytological images," IEEE Systems Journal, vol. 8, no. 3, pp. 949–964, 2014.

ДОДАТОК А ЛІСТІНГ ПРОГРАМИ

```
import tensorflow as tf
import numpy as np
from scipy.misc import imread, imresize
from keras.applications.vgg16 import VGG16
from keras.preprocessing import image
from keras.applications.vgg16 import preprocess_input, decode_predictions
#from imagenet_classes import class_names
import os

class vgg16:
    def __init__(self, imgs, weights=None, sess=None):
        self.imgs = imgs
        self.convlayers()
        self.fc_layers()
        self.probs = tf.nn.softmax(self.fc3l)
        if weights is not None and sess is not None:
            self.load_weights(weights, sess)
    def convlayers(self):
        self.parameters = []
        # zero-mean input
        with tf.name_scope('preprocess') as scope:
            mean = tf.constant([123.68, 116.779, 103.939], dtype=tf.float32, shape=[1, 1, 1, 3], name='img_mean')
            images = self.imgs-mean
        # conv1_1
        with tf.name_scope('conv1_1') as scope:
            kernel = tf.Variable(tf.truncated_normal([3, 3, 3, 64], dtype=tf.float32,stddev=1e-1), name='weights')
            conv = tf.nn.conv2d(images, kernel, [1, 1, 1, 1], padding='SAME')
            biases = tf.Variable(tf.constant(0.0, shape=[64], dtype=tf.float32),trainable=True, name='biases')
            out = tf.nn.bias_add(conv, biases)
            self.conv1_1 = tf.nn.relu(out, name=scope)
            self.parameters += [kernel, biases]
        # conv1_2
        with tf.name_scope('conv1_2') as scope:
            kernel = tf.Variable(tf.truncated_normal([3, 3, 64, 64], dtype=tf.float32,stddev=1e-1), name='weights')
            conv = tf.nn.conv2d(self.conv1_1, kernel, [1, 1, 1, 1], padding='SAME')
            biases = tf.Variable(tf.constant(0.0, shape=[64], dtype=tf.float32),trainable=True, name='biases')
            out = tf.nn.bias_add(conv, biases)
            self.conv1_2 = tf.nn.relu(out, name=scope)
            self.parameters += [kernel, biases]
        # pool1
        self.pool1 = tf.nn.max_pool(self.conv1_2,ksize=[1, 2, 2, 1],strides=[1, 2, 2, 1],padding='SAME',name='pool1')
        # conv2_1
        with tf.name_scope('conv2_1') as scope:
            kernel = tf.Variable(tf.truncated_normal([3, 3, 64, 128], dtype=tf.float32,stddev=1e-1), name='weights')
            conv = tf.nn.conv2d(self.pool1, kernel, [1, 1, 1, 1], padding='SAME')
            biases = tf.Variable(tf.constant(0.0, shape=[128], dtype=tf.float32),trainable=True, name='biases')
            out = tf.nn.bias_add(conv, biases)
            self.conv2_1 = tf.nn.relu(out, name=scope)
            self.parameters += [kernel, biases]
        # conv2_2
        with tf.name_scope('conv2_2') as scope:
            kernel = tf.Variable(tf.truncated_normal([3, 3, 128, 128], dtype=tf.float32,stddev=1e-1), name='weights')
            conv = tf.nn.conv2d(self.conv2_1, kernel, [1, 1, 1, 1], padding='SAME')
            biases = tf.Variable(tf.constant(0.0, shape=[128], dtype=tf.float32),trainable=True, name='biases')
            out = tf.nn.bias_add(conv, biases)
            self.conv2_2 = tf.nn.relu(out, name=scope)
            self.parameters += [kernel, biases]
        # pool2
        self.pool2 = tf.nn.max_pool(self.conv2_2,ksize=[1, 2, 2, 1],strides=[1, 2, 2, 1],padding='SAME',name='pool2')
        # conv3_1
        with tf.name_scope('conv3_1') as scope:
            kernel = tf.Variable(tf.truncated_normal([3, 3, 128, 256], dtype=tf.float32,stddev=1e-1), name='weights')
            conv = tf.nn.conv2d(self.pool2, kernel, [1, 1, 1, 1], padding='SAME')
            biases = tf.Variable(tf.constant(0.0, shape=[256], dtype=tf.float32),trainable=True, name='biases')
            out = tf.nn.bias_add(conv, biases)
            self.conv3_1 = tf.nn.relu(out, name=scope)
            self.parameters += [kernel, biases]
        # conv3_2
        with tf.name_scope('conv3_2') as scope:
            kernel = tf.Variable(tf.truncated_normal([3, 3, 256, 256], dtype=tf.float32,
                                                     stddev=1e-1), name='weights')
            conv = tf.nn.conv2d(self.conv3_1, kernel, [1, 1, 1, 1], padding='SAME')
            biases = tf.Variable(tf.constant(0.0, shape=[256], dtype=tf.float32),
                                 trainable=True, name='biases')
            out = tf.nn.bias_add(conv, biases)
            self.conv3_2 = tf.nn.relu(out, name=scope)
            self.parameters += [kernel, biases]
        # conv3_3
        with tf.name_scope('conv3_3') as scope:
            kernel = tf.Variable(tf.truncated_normal([3, 3, 256, 256], dtype=tf.float32,stddev=1e-1), name='weights')
            conv = tf.nn.conv2d(self.conv3_2, kernel, [1, 1, 1, 1], padding='SAME')
            biases = tf.Variable(tf.constant(0.0, shape=[256], dtype=tf.float32),trainable=True, name='biases')
            out = tf.nn.bias_add(conv, biases)
            self.conv3_3 = tf.nn.relu(out, name=scope)
            self.parameters += [kernel, biases]
        # pool3
        self.pool3 = tf.nn.max_pool(self.conv3_3,ksize=[1, 2, 2, 1],strides=[1, 2, 2, 1],padding='SAME',name='pool3')
        # conv4_1
        with tf.name_scope('conv4_1') as scope:
            kernel = tf.Variable(tf.truncated_normal([3, 3, 256, 512], dtype=tf.float32,stddev=1e-1), name='weights')
            conv = tf.nn.conv2d(self.pool3, kernel, [1, 1, 1, 1], padding='SAME')
            biases = tf.Variable(tf.constant(0.0, shape=[512], dtype=tf.float32),trainable=True, name='biases')
            out = tf.nn.bias_add(conv, biases)
            self.conv4_1 = tf.nn.relu(out, name=scope)
            self.parameters += [kernel, biases]
        # conv4_2
        with tf.name_scope('conv4_2') as scope:
```

```
            kernel = tf.Variable(tf.truncated_normal([3, 3, 512, 512], dtype=tf.float32,stddev=1e-1), name='weights')
            conv = tf.nn.conv2d(self.conv4_1, kernel, [1, 1, 1, 1], padding='SAME')
            biases = tf.Variable(tf.constant(0.0, shape=[512], dtype=tf.float32),trainable=True, name='biases')
            out = tf.nn.bias_add(conv, biases)
            self.conv4_2 = tf.nn.relu(out, name=scope)
            self.parameters += [kernel, biases]
        # conv4_3
        with tf.name_scope('conv4_3') as scope:
            kernel = tf.Variable(tf.truncated_normal([3, 3, 512, 512], dtype=tf.float32,stddev=1e-1), name='weights')
            conv = tf.nn.conv2d(self.conv4_2, kernel, [1, 1, 1, 1], padding='SAME')
            biases = tf.Variable(tf.constant(0.0, shape=[512], dtype=tf.float32),trainable=True, name='biases')
            out = tf.nn.bias_add(conv, biases)
            self.conv4_3 = tf.nn.relu(out, name=scope)
            self.parameters += [kernel, biases]
        # pool4
        self.pool4 = tf.nn.max_pool(self.conv4_3,ksize=[1, 2, 2, 1],strides=[1, 2, 2, 1],padding='SAME',name='pool4')
        # conv5_1
        with tf.name_scope('conv5_1') as scope:
            kernel = tf.Variable(tf.truncated_normal([3, 3, 512, 512], dtype=tf.float32,stddev=1e-1), name='weights')
            conv = tf.nn.conv2d(self.pool4, kernel, [1, 1, 1, 1], padding='SAME')
            biases = tf.Variable(tf.constant(0.0, shape=[512], dtype=tf.float32),trainable=True, name='biases')
            out = tf.nn.bias_add(conv, biases)
            self.conv5_1 = tf.nn.relu(out, name=scope)
            self.parameters += [kernel, biases]
        # conv5_2
        with tf.name_scope('conv5_2') as scope:
            kernel = tf.Variable(tf.truncated_normal([3, 3, 512, 512], dtype=tf.float32,stddev=1e-1), name='weights')
            conv = tf.nn.conv2d(self.conv5_1, kernel, [1, 1, 1, 1], padding='SAME')
            biases = tf.Variable(tf.constant(0.0, shape=[512], dtype=tf.float32),trainable=True, name='biases')
            out = tf.nn.bias_add(conv, biases)
            self.conv5_2 = tf.nn.relu(out, name=scope)
            self.parameters += [kernel, biases]
        # conv5_3
        with tf.name_scope('conv5_3') as scope:
            kernel = tf.Variable(tf.truncated_normal([3, 3, 512, 512], dtype=tf.float32,stddev=1e-1), name='weights')
            conv = tf.nn.conv2d(self.conv5_2, kernel, [1, 1, 1, 1], padding='SAME')
            biases = tf.Variable(tf.constant(0.0, shape=[512], dtype=tf.float32),trainable=True, name='biases')
            out = tf.nn.bias_add(conv, biases)
            self.conv5_3 = tf.nn.relu(out, name=scope)
            self.parameters += [kernel, biases]
        # pool5
        self.pool5 = tf.nn.max_pool(self.conv5_3,ksize=[1, 2, 2, 1],strides=[1, 2, 2, 1],padding='SAME',name='pool4')
    def fc_layers(self):
        # fc1
        with tf.name_scope('fc1') as scope:
            shape = int(np.prod(self.pool5.get_shape()[1:]))
            fc1w = tf.Variable(tf.truncated_normal([shape, 4096],dtype=tf.float32,stddev=1e-1), name='weights')
            fc1b = tf.Variable(tf.constant(1.0, shape=[4096], dtype=tf.float32),trainable=True, name='biases')
            pool5_flat = tf.reshape(self.pool5, [-1, shape])
            fc1l = tf.nn.bias_add(tf.matmul(pool5_flat, fc1w), fc1b)
            self.fc1 = tf.nn.relu(fc1l)
            self.parameters += [fc1w, fc1b]
        # fc2
        with tf.name_scope('fc2') as scope:
            fc2w = tf.Variable(tf.truncated_normal([4096, 4096],dtype=tf.float32,stddev=1e-1), name='weights')
            fc2b = tf.Variable(tf.constant(1.0, shape=[4096], dtype=tf.float32),trainable=True, name='biases')
            fc2l = tf.nn.bias_add(tf.matmul(self.fc1, fc2w), fc2b)
            self.fc2 = tf.nn.relu(fc2l)
            self.parameters += [fc2w, fc2b]
        # fc3
        with tf.name_scope('fc3') as scope:
            fc3w = tf.Variable(tf.truncated_normal([4096, 1000],dtype=tf.float32,stddev=1e-1), name='weights')
            fc3b = tf.Variable(tf.constant(1.0, shape=[1000], dtype=tf.float32),trainable=True, name='biases')
            self.fc3l = tf.nn.bias_add(tf.matmul(self.fc2, fc3w), fc3b)
            self.parameters += [fc3w, fc3b]
    def load_weights(self, weight_file, sess):
        weights = np.load(weight_file)
        keys = sorted(weights.keys())
        for i, k in enumerate(keys):
            print(i, k, np.shape(weights[k]))
            sess.run(self.parameters[i].assign(weights[k]))

if __name__ == '__main__':
    sess = tf.Session()
    imgs = tf.placeholder(tf.float32, [None, 224, 224, 3])

    #model = VGG16(include_top=False, weights='imagenet')

    #model.save_weights()

    vgg = vgg16(imgs, 'vgg16_weights.npz', sess)
    #running for classification folder
    i=0
    for file in os.listdir('/Users/igorvarga/PycharmProjects/classifying-cancer/cnn_image_classifier/images/predict/benign/'):
        if file.endswith('.png'):
            name=os.path.join('/Users/igorvarga/PycharmProjects/classifying-cancer/cnn_image_classifier/images/predict/benign/',file)
            base_name = name[:-4]
            img1 = imread(name)
            img1 = imresize(img1, (224, 224))
        #img1 = img1[:,:,0:-1]
            prob = sess.run(vgg.fc2, feed_dict={vgg.imgs: [img1]})[0]
            np.save(base_name,prob)
            print(len(prob))
            i = i+1
            #for x in prob:
             #   if x == 0.0:
              #      i=i+1
            print(i)

import math


class FuzzyNeuralNetwork(object):
   conj = None
```

```
    out_f = None
    """docstring for FuzzyNeuralNetwork"""
    def __init__(self, rules):
        super(FuzzyNeuralNetwork, self).__init__()
        self.rules = rules
    def __init__(self, rules,conjunct_func,out_func):
        super(FuzzyNeuralNetwork, self).__init__()
        self.rules = rules
        self.conj = conjunct_func
        self.out_f = out_func

    def activate(self, vec, fn=None,fo=None):
        if fn is None:
            return self.activate(vec,self.conj,self.out_f)
        res = []
        for i in range(len(self.rules)):
            res.append(self.rules[i](vec,fn))

        tres = []
        for x in zip(*res) :
            tv = 0.0
            for y in x :
                tv = fo(tv,y)
            tres.append(tv)
        return tres


    def mape(self, test):
        tmp = 0
        for i in range(len(test)):
            v = test[i]
            clas = v[1]
            x = self.activate(v[0],self.conj,self.out_f)
            v = max(x)
            c = x.index(v)
            if c == clas:
                tmp += 1

        return tmp/len(test)
    #def activate2(self, vec):
    #  return self.activate(vec,self.conj,self.out_f)

class Rule(object):
    """docstring for Rule"""
    def __init__(self, inp_w, out_w = None,cls = None):
        super(Rule, self).__init__()
        self.inp_weights = inp_w
        self.out_weights = out_w
        self.cl = cls

    def __eq__(self, other):
        a = True
        if type(other) != type(self):
            return False
        if len(self.inp_weights) != len(other.inp_weights):
            return False
        for i in range(len(self.inp_weights)):
            a = a and (self.inp_weights[i] == other.inp_weights[i])

        for i in range(len(self.out_weights)):
            a = a and (self.out_weights[i] == other.out_weights[i])
        return a


    def __call__(self, *args, **kwargs):
        """

        :param args: 1 - vector of input 2 - conjunction func
        :param kwargs:
        :return: vector output
        """
        t_val  = []
        if len(args[0]) != len (self.inp_weights) :
            raise Exception("lengts is not valid")
        for x in range(0,len(args[0])):
            t_val.append( self.inp_weights[x]( args[0][x] ) )
        #conjunction
        t_val1 = 1.0
        for x in range(0, len(t_val)):
            t_val1 = args[1](t_val[x],t_val1)

        t_val = [0.0] * len(self.out_weights)


        for x in range(0,len(self.out_weights)):
            t_val[x] = self.out_weights[x] * t_val1
        return t_val

    def activate1(self, *args):
        """

        :param args: 1 - vector of input 2 - conjunction func
        :param kwargs:
        :return: vector output
        """
        t_val  = []
        if len(args[0]) != len (self.inp_weights) :
            raise Exception("lengts is not valid")
        for x in range(0,len(args[0])):
            t_val.append( self.inp_weights[x]( args[0][x] ) )
        #conjunction
        t_val1 = 1.0
        for x in range(0, len(t_val)):
            t_val1 = args[1](t_val[x],t_val1)
```

```
      return t_val1
      pass


class Gauss(object):
   """docstring for Gauss"""
   def __init__(self, a, b):
      super(Gauss, self).__init__()
      self.a = a
      self.b = b

   def __call__(self, *args):
      x = args[0]
      return math.exp(0 - ((x - self.a)*(x - self.a)/(self.b * self.b)))

   def __repr__(self):
      return "a : " + str(self.a) + " b: " + str(self.b)

   def __eq__(self, other):
      a = True
      if type(other) != type(self):
         return False
      a = a and (self.a == other.a) and (self.b == other.b)
      return a


def evaluate_interval(train_set):
   interv = []

   for x in (zip(*train_set)):
      lv = [min(x), max(x)]
      interv.append(lv)
   return interv


if __name__ == '__main__':
   lala = Rule([lambda x: x*x, lambda x: x*x*x],[123])
   lala([12.132434,11.4345],lambda x,y: x*y)
   import copy
   a = 4
   ala = Gauss(a,1)
   ala2 = copy.copy(ala)
   a = 5
   print(ala2)
   #print lala.inp_weights
      class StandardDeviationNefclassAdditive():
   fnn = None
   desc = []
   rule_classes = []
   t_vals = []

   def initiate(self, descr, rul_cls, trn):
      self.desc = descr
      self.rule_classes = rul_cls
      self.t_vals = trn
      self.cnt = 0


   def restoreFNN(self,weights):
      self.cnt = self.cnt+1
      W = weights
      wrc = W[0-len(self.rule_classes):]
      wgts = W[:0-len(self.rule_classes)]
      wgts = list(chunks(wgts,int(len(wgts)/len(self.rule_classes))))

      rules = []
      for i in range(len(self.rule_classes)):

         rc = self.rule_classes[i]

         inter = wgts[i]
         b_s = inter[int(len(inter)/2):]
         a_s = inter[:int(len(inter)/2)]
         gausses = [ns.Gauss(a_s[x],b_s[x]) for x in range(len(a_s))]
         ow = [0.0] * self.desc[0]
         ow[rc] = wrc[i]
         rule = ns.Rule(gausses,ow)
         rule.rc = rc
         rules.append(rule)

      con = lambda x,y : x*y
      out_fu = lambda x,y : x+y
      fnn  = ns.FuzzyNeuralNetwork(rules,con,out_fu)
      return fnn



      pass
   def __call__(self, *args, **kwargs):
      self.cnt = self.cnt+1
      W = args[0]
      wrc = W[0-len(self.rule_classes):]
      wgts = W[:0-len(self.rule_classes)]
      wgts = list(chunks(wgts,int(len(wgts)/len(self.rule_classes))))

      rules = []
      for i in range(len(self.rule_classes)):

         rc = self.rule_classes[i]

         inter = wgts[i]
         b_s = inter[int(len(inter)/2):]
         a_s = inter[:int(len(inter)/2)]
         gausses = [ns.Gauss(a_s[x],b_s[x]) for x in range(len(a_s))]
```

```
            ow = [0.0] * self.desc[0]
            ow[rc] = wrc[i]
            rule = ns.Rule(gausses,ow)
            rule.rc = rc
            rules.append(rule)

        con = lambda x,y : x*y
        out_fu = lambda x,y : x+y
        fnn  = ns.FuzzyNeuralNetwork(rules,con,out_fu)


        result = 0.0
        #k = fnn.mape(train)
        #j = fnn.mape(test)
        #print(j)
        for i in range(len(self.t_vals)):
            vec = self.t_vals[i]
            t_res = fnn.activate(vec[0])

            for j in range(len(t_res)):
                if j == vec[1]:
                    result = result + (1 - t_res[j])*(1 - t_res[j])
                else:
                    result = result + (0 - t_res[j]) * (0 - t_res[j])
                pass

        return result/len(self.t_vals)



def train_weights():
    global t_rules
    import pickle

    fi = open('/Users/igorvarga/nefclass-jupyter/fo.out','rb')
    try:
        unpickler = pickle.Unpickler(fi)
        t_rules = unpickler.load()

    except EOFError:
        return
    #t_rules =     pickle.load(fi)
    #t_rules = list(t_rules)
    global cl_cnt
    global db
    global dc
    global test
    #set output weight
    #

    wins = []
    wouts = []
    r_classes = []
    for i in range(len(t_rules)):
        v = t_rules[i]
        a_s = []
        b_s = []
        for j in range(len(v.inp_weights)):
            g = v.inp_weights[j]
            a_s.append(g.a)
            b_s.append(g.b)
        wins += a_s
        wins += b_s
        wouts.append(v.out_weights[v.cl])
        r_classes.append(v.cl)

    devi = StandardDeviationNefclassAdditive()

    devi.initiate([cl_cnt],r_classes,train)
    x0 = wins + wouts
    import scipy.optimize as opt
    res = opt.fmin_cg(devi,x0,maxiter=4000,gtol=0.003)
    resfnn = devi.restoreFNN(res)
    print("" + str(resfnn.mape(train)) + "," + str(resfnn.mape(test)))
    print("ddffkeffddfv")


def grad_func():
    pass



def train_rules():

    global inp_dim
    global perc
    global cl_cnt
    global train
    global test
    global training_algorithm
    global kmax
    global t_rules

    #### TRAIN_RULES



    inp_dim = len(train[0][0])
    ts = [x[0] for x in train]
    interv = evaluate_interval(ts)
    t_interv = copy.deepcopy(interv)
    fir_gauss = evaluate_gauss_first(t_interv)
```

```
    antedescents = []

    for i in range(len(train)):
        cc = train[i][1]
        rule = create_antedescent(fir_gauss,train[i][0])
        ow = [0.0] * cl_cnt
        ow[cc] = 1.0
        rule.out_weights = ow

        if rule in antedescents:
            pass
        else:
            rule.cl = cc
            antedescents.append(rule)

    #activation count per rule
    acc = [[0.0] * cl_cnt for x in range(len(antedescents))]

    for i in range(len(train)):
        cc = train[i][1]
        print(i)
        for j in range(len(antedescents)):
            x = acc[j][cc]
            x = x + antedescents[j].activate1(train[i][0],mult)
            acc[j][cc] = x
    print(len(antedescents))

    #evaluate performance
    perf = []
    for i in range(len(antedescents)):
        ant = acc[i]
        tmp_r = antedescents[i]
        t = max(ant)
        #antedescents[i].cl = ant.index(t)
        rs = ant[tmp_r.cl]
        for j in range(len(ant)):
            if j != tmp_r.cl:
                rs = rs - ant[j]
        perf.append(rs)

    #rules T
    t_rules = []

    if training_algorithm == "BEST_PER_CLASS":
        ppc = [[] for x in range(cl_cnt)]
        rpc = [[] for x in range(cl_cnt)]
        for i in range(len(antedescents)):
            r = antedescents[i]
            rc = r.cl
            rpc[rc].append(r)
            ppc[rc].append(perf[i])

        for i in range(cl_cnt):
            p = ppc[i]
            r = rpc[i]

            for j in range(int(kmax //cl_cnt)):
                v = max(p)
                x = p.index(v)
                t_rules.append(r[x])

                p.pop(x)
                r.pop(x)

        pass
    if training_algorithm == "BEST":

        pass
    #print(fir_gauss)

    #return t_rules

    #print(num_of_weights)
    #return 43
    #fnn = ns.FuzzyNeuralNetwork(rules=21)
    pass

def evaluate_interval(data):
    interv = []
    #print(data)
    for x in (zip(*data)):
        lv = [min(x), max(x)]
        interv.append(lv)
    return interv

def evaluate_gauss_first(interv):
    global inp_dim
    global num_of_weights
    res = []
    for i in range(0, inp_dim):
        tl = []
        delta = (interv[i][1] - interv[i][0]) / (num_of_weights - 1)
        for j in range(0,num_of_weights):
            g = ns.Gauss(interv[i][0] + delta * j, delta)
            tl.append(g)
        res.append(tl)
    return res

    pass

def create_antedescent(v_gauss,inp):
    vg = []
    for i in range(0, len(inp)):
        tmp = []
```

```
      for j in range(0, len(v_gauss[i])):
         tmp.append(copy.copy(v_gauss[i][j]))
      vec = find_max_in_weights(tmp,inp[i])
      vg.append(vec)

   vg = ns.Rule(vg)
   return vg


gr = (math.sqrt(5) + 1) / 2

def goldensection_method(f, a, b, tol=1e-5):

   c = b - (b - a) / gr
   d = a + (b - a) / gr
   while abs(c - d) > tol:
      if f(c) < f(d):
         b = d
      else:
         a = c

      # we recompute both c and d here to avoid loss of precision which may lead to incorrect results or infinite loop
      c = b - (b - a) / gr
      d = a + (b - a) / gr

   return (b + a) / 2

def find_max_in_weights(vec_gauss, x):
   max = 0
   maxtmp = vec_gauss[0](x)
   for i in range(len(vec_gauss)):
      if i > 0:
         tmp_v = vec_gauss[i](x)
         if tmp_v > maxtmp:
            maxtmp = tmp_v
            max = copy.copy(i)
   return copy.copy(vec_gauss[max])


if __name__ == '__main__':
   i = 2
   j = i
   i = i + 1
   print(j)
   td = file_reader("/Users/igorvarga/Documents/WBC/wdbc.data.txt")
   train_set = td[0]
   lv = evaluate_interval(td[0])
   print(lv)
```

ДОДАТОК Б ІЛЮСТРАТИВНІ МАТЕРІАЛИ

Дякую за увагу

Висновки

В ході її виконання були отримані наступні науково-практичні результати:

1.Використання попередньо навченої моделі підходить для визначення типу клітини, проте не підходить для визначення підтипу

2.Експерименти показали, що із проаналізованих варіантів та порівнянь зі зробленими попередньо конфігураціями мереж, найкраще результати показала мережа нефклас

Також можливі наступні варіанти розвитку данного продукту, а саме:

1.Використати інші алгоритми навчання

2.Використати навчання на відеокарті для оптимізації швидкості

3.Використати інші види класифікаторів

4.Спробувати використати інші види декомпозиції

5.Спробувати підкоригувати систему ваг на мережі VGG-16, тобто перенавчити її, що допоможе класифікувати данні більш точно, а саме визначати підтип пухлини

Table 1-1 порівняння використання різних класифікаторів

	40X	100X	200X	400X
Linear svm	89%	89%	88%	88%
Polynomial svm	88%	90%	89%	85%
Random forest	89.18%	88%	87.74%	80%
Nefclass	90%	91%	90.5%	90%

Table 1-2Таблиця залежності варіації від к-сті компонент

Кількість компонент	Варіація	Час навчання(приблизний)
100	0.840587442159	~2 год
200	0.897366730496	~3 год
250	0.912324353994	~4 год
500	0.954868534936	~9 год

Table 1-3 Точність експерименту при різних кількостях признаків

Початкове розбиття,кількість правил	100	250	4096
4,4	75.23%	80.64%	76.3%
4,6	83.34%	87.24%	91%
4,8	84.21%	88.18%	89.8%

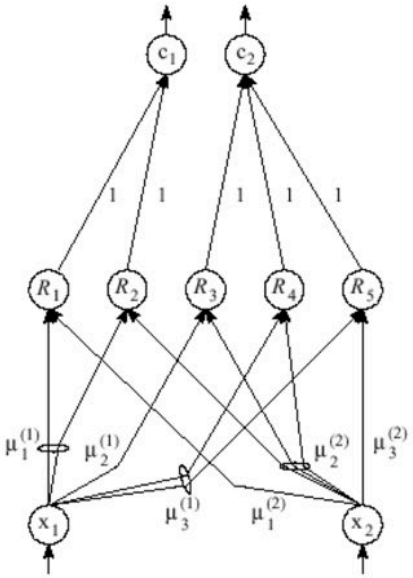
Результати

Початкове розбиття,кількість правил	40X	100X	200X	400X
2,2	73%	74%	74.2%	73.5%
4,2	75.3%	74.8%	75.7%	75.4%
6,2	78.2%	79%	78.4%	78%
8,2	76%	75.4%	76.5%	75.8%
2,4	75%	74%	73.8%	73%
4,4	78.3%	76.3%	75.7%	75.4%
6,4	82%	83%	82.4%	83.2%
8,4	82.2%	81.5%	81.5%	83.8%
2,6	75.4%	73.8%	74.4%	73.2%
4,6	**90%**	**91%**	**90.5%**	**90%**
6,6	89%	89.7%	90.2%	89.5%
8,6	90.3%	90.5%	92%	91.2%
4,8	89.3%	89.8%	89.7%	89.3%
6,8	89.2%	88%	89.4%	88.4%
8,8	88%	87.2%	87.2%	87%

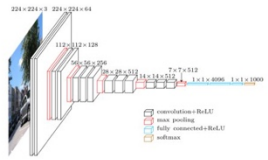
Підхід навчання та обробки класифікатора

- Порядок обробки даних:
  1. Зжати вхідне зображення до 224x224
  2. Подати на натреновану модель VGG-16 з видаленим останнім шаром кожне зображення
  3. Обробити признаки
  4. Подати признаки на класифікуючий алгоритм(NefClass)

Нечітка нейронна мережа NefClass



- Навчання правил в експерименті бралось найкраще за клас
- В якости нечіткої ваги бралась функція eˆ((- (x-a)ˆ2)/bˆ2))



- ImageNet
- 1.2 млн зображень
- 1000 класів

Перенос навчання

- Є два основних сценарії переносу навчання:

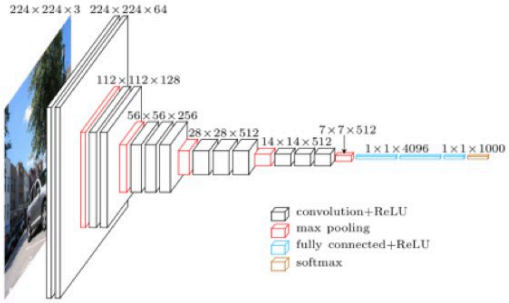
  1. Виділення признаків
  У цьому випадку останній повнозвязний шар видаляється, а решту CNN використовується як екстрактор для нового набору даних.

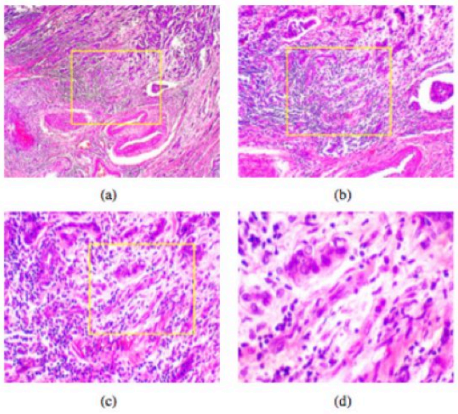
  2. Точна настройка
  Тут новий набір даних використовується для точного налаштування ваг попередньо підготовленої CNN. Можна точно налаштувати всі шари або навіть певні шари CNN.

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	(None, 224, 224, 3)	0	
block1_conv1 (Convolution2D)	(None, 224, 224, 64)	1792	input_1[0][0]
block1_conv2 (Convolution2D)	(None, 224, 224, 64)	36928	block1_conv1[0][0]
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0	block1_conv2[0][0]
block2_conv1 (Convolution2D)	(None, 112, 112, 128)	73856	block1_pool[0][0]
block2_conv2 (Convolution2D)	(None, 112, 112, 128)	147584	block2_conv1[0][0]
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0	block2_conv2[0][0]
block3_conv1 (Convolution2D)	(None, 56, 56, 256)	295168	block2_pool[0][0]
block3_conv2 (Convolution2D)	(None, 56, 56, 256)	590080	block3_conv1[0][0]
block3_conv3 (Convolution2D)	(None, 56, 56, 256)	590080	block3_conv2[0][0]
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0	block3_conv3[0][0]
block4_conv1 (Convolution2D)	(None, 28, 28, 512)	1180160	block3_pool[0][0]
block4_conv2 (Convolution2D)	(None, 28, 28, 512)	2359808	block4_conv1[0][0]
block4_conv3 (Convolution2D)	(None, 28, 28, 512)	2359808	block4_conv2[0][0]
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0	block4_conv3[0][0]
block5_conv1 (Convolution2D)	(None, 14, 14, 512)	2359808	block4_pool[0][0]
block5_conv2 (Convolution2D)	(None, 14, 14, 512)	2359808	block5_conv1[0][0]
block5_conv3 (Convolution2D)	(None, 14, 14, 512)	2359808	block5_conv2[0][0]
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0	block5_conv3[0][0]
flatten (Flatten)	(None, 25088)	0	block5_pool[0][0]
fc1 (Dense)	(None, 4096)	102764544	flatten[0][0]
fc2 (Dense)	(None, 4096)	16781312	fc1[0][0]
predictions (Dense)	(None, 1000)	4097000	fc2[0][0]

Total params: 138357544

Згорткові нейронні мережі





- Слайд злоякісної пухлини молочної залози (забарвленої) спостерігається в різних факторах збільшення: (a) 40X, (b) 100X, (c) 200X та (d) 400X.

Масштаб	Доброякісні	Злоякісні	Загально
40X	652	1370	1995
100X	644	1437	2081
200X	623	1390	2013
400X	588	1232	1820
Загальна кількість зображень	2480	5429	7909

BreakHis

- Зображення були відібрані в ході клінічного дослідження з січня 2014 року по грудень 2014 року
- BreaKHis складається з 7909 клінічно репрезентативних мікроскопічних зображень грудних опухолей зібраних у 82 пацієнтів з різними масштабами (40X, 100X, 200X, 400X)
- База містить 2,480 доброякісних і 5429 злоякісних зразків (700X460 пікселів, 3-канальний RGB, 8-бітна глибина в кожному каналі, формат PNG).

4

Актуальність

- Близько 1 із 8 жінок (близько 12,4%) мають інвазивний рак молочної залози протягом усього життя
- Очікується, що у чоловіків в 2018 році діагностується близько 2550 нових випадків інвазивного раку молочної залози. Ризику раку молочної залози у чоловіка складає приблизно 1 з 1000
- Для жінок у США смертність від раку молочної залози вище, ніж у будь-якого іншого раку, крім раку легенів.
- Крім раку шкіри, рак молочної залози є найбільш часто діагностованим раком серед американських жінок. У 2017 році, за оцінками, близько 30% випадків раннього діагностування ракових захворювань у жінок стануть раком молочної залози.

3

Тема роботи

- Об’єкт дослідження: медичні зображення тканей людини
- Предмет дослідження: методи розпізнавання медичних зображень та діагностики

2

Розпізнавання медичних зображень з використанням нечітких нейронних мереж

1